AN OPTIMAL PURSUIT-EVASION STRATEGY FOR LINEAR SAMPLED-DATA SYSTEMS

CHARLES HOWARD HUTCHISON
and
LAWRENCE FLANDERS PERMENTER

AN OPTIMAL PURSUIT-EVASION STRATEGY FOR LINEAR SAMPLED-DATA SYSTEMS

by

Charles Howard Hutchison
Lieutenant, United States Navy
B.S., Duke University, 1959

and

Lawrence Flanders Permenter
Lieutenant, United States Navy
B.S., United States Naval Academy, 1959

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 1967

ABSTRACT

Differential game theory and dynamic programming are applied to derive the optimal strategy, or sequence of controls, for a class of linear, sampled-data, pursuit-evasion problems. The necessary and sufficient conditions for existence of the solution are derived. A digital computer program for simulating control generation and system trajectories is given. The results of simulation tests using this digital computer model are presented to demonstrate the salient characteristics of the strategy.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

1. INTRODUCTION

Recent literature in the field of automatic control theory shows an increasing interest in the application of differential game theory to control system design [1, 2, 3]. This approach to control system design may be considered an extension of "optimal" control theory allowing one or more opposing systems controlled by intelligent opponents to be introduced into the design computations. ("Intelligent" is used here to mean having knowledge of the opponent's system dynamics, states, and desires.) Further, this approach assumes that the controller of each system will act in accordance with his intelligence. This allows system design to provide satisfactory performance when an opponent acts to minimize that performance, and to provide improved performance when the opponent acts otherwise.

The disadvantages of this approach to control system design are generally those attributed to optimal control theory in the past --- the difficulty of expressing realistic performance objectives in the mathematical form of a performance index, and the difficulty of obtaining solutions to problems with realistic physical constraints. These difficulties are somewhat increased by the introduction of intelligent opponents.

2. STATEMENT OF THE PROBLEM

This research was confined to exploring the solution and salient characteristics of the sampled-data analog of a continuous system pursuit-evasion problem and solution published by Y.C. Ho, A.E. Bryson, and S. Baron [3].

2.1 Differential Constraints

The pursuit and evasion plants are described by a set of linear, time-invariant, first-order state equations [4]:

$$\dot{\underline{x}}_p = A_p \underline{x}_p + B_p \underline{u} \tag{1}$$

$$\dot{\underline{x}}_e = A_e \underline{x}_e + B_e \underline{v} \quad . \tag{2}$$

The following definitions apply to equations (1) and (2):

$\underline{x}_p$ is the (n by 1) state vector of the pursuit plant

$\underline{x}_e$ is the (n by 1) state vector of the evasion plant

$\underline{u}$ is the (m_p by 1) control vector of the pursuit plant

$\underline{v}$ is the (m_e by 1) control vector of the evasion plant

A_p and A_e are (n by n) matrices of constant coefficients

B_p is an (n by m_p) matrix of constant coefficients

B_e is an (n by m_e) matrix of constant coefficients.

Since the controls $\underline{u}$ and $\underline{v}$ are to be applied in sampled-data form, it is convenient to describe the plants by the following difference constraint equations [4]:

$$\underline{x}_p(k+1) = \phi_p(T)\underline{x}_p(k) + \Delta_p(T)\underline{u}(k) \tag{3}$$

$$\underline{x}_e(k+1) = \phi_e(T)\underline{x}_e(k) + \Delta_e(T)\underline{v}(k) \tag{4}$$

where ϕ_p and ϕ_e are state transition matrices defined by:

$$\phi_p(T) = e^{A_p T} \tag{5}$$

$$\phi_e(T) = e^{A_e T} \quad \text{where,} \tag{6}$$

$$e^{AT} = I + \frac{AT}{1!} + \frac{A^2T^2}{2!} + \frac{A^3T^3}{3!} + \ldots \tag{7}$$

The state distribution matrices are defined:

$$\Delta_p(T) = \int_0^T \phi_p(T-t)B_p dt \tag{8}$$

$$\Delta_e(T) = \int_0^T \phi_e(T-t)B_e dt \quad . \tag{9}$$

The integrals (8) and (9) can be evaluated by the infinite series:

$$\Delta(T) = \left[IT + \frac{AT^2}{2!} + \frac{A^2T^3}{3!} + \frac{A^3T^4}{4!} + \ldots \right] B \tag{10}$$

2.2 Performance Index

The performance index is defined:

$$J = [\underline{x}_p(t_f) - \underline{x}_e(t_f)]^T Q[\underline{x}_p(t_f) - \underline{x}_e(t_f)]$$

$$+ \int_0^{t_f} [\underline{u}^T(t)R_p(t)\underline{u}(t) - \underline{v}^T(t)R_e\underline{v}(t)]dt \quad . \tag{11}$$

The following additional definitions apply to the performance index:

Q is a (n by n) positive semi-definite, symmetric, final-state-difference weighting matrix.

R_p is a (m_p by m_p) positive definite, symmetric, pursuit control weighting matrix.

R_e is a (m_e by m_e) positive definite, symmetric, evasion control weighting matrix.

Since the controls are to be applied in sampled-data form ($\underline{u}$ and $\underline{v}$ constant for the sample period T) for N sample periods ($t_f = NT$), the performance index given in (11) is represented by:

$$J = [\underline{x}_p(N) - \underline{x}_e(N)]^T Q[\underline{x}_p(N) - \underline{x}_e(N)]$$

$$+ T \sum_{k=0}^{N-1} [\underline{u}^T(k)R_p(k)\underline{u}(k) - \underline{v}^T(k)R_e(k)\underline{v}(k)] \quad . \tag{12}$$

The desired optimal solution is a set of controls:

$$\underline{u}^o = \{\underline{u}(0), \underline{u}(1), \ldots \underline{u}(N-1)\} \quad \text{to minimize J,}$$

and

$$\underline{v}^o = \{\underline{v}(0), \underline{v}(1), \ldots \underline{v}(N-1)\} \quad \text{to maximize J.}$$

Equivalently, the saddle point

$$J(\underline{u}^o, \underline{v}) \leq J(\underline{u}^o, \underline{v}^o) \leq J(\underline{u}, \underline{v}^o) \qquad \text{is sought.} \tag{13}$$

It is assumed at this point that such a saddle point does exist and that the optimal value of the performance index is:

$$J^o(\underline{u},\underline{v}) = J(\underline{u}^o,\underline{v}^o) = \min_{\underline{u}} \left[\max_{\underline{v}} J \right] = \max_{\underline{v}} \left[\min_{\underline{u}} J \right] \tag{14}$$

The dominant characteristics of this performance index are as follows:

1) The form of the performance index requires that the orders of the pursuit and the evasion plants be the same (otherwise $\underline{x}_p$ and $\underline{x}_e$ would not be conformable for subtraction). The dimensions of the pursuit and evasion control vectors, however, need not be the same (m_p need not equal m_e).

2) The final state difference weighting matrix Q is chosen positive <u>semi-definite</u> in order to reflect an increase in the performance index for an increase in the difference of the <u>states of interest only</u> of the two plants at the final time.

3) The R_p matrix is chosen positive definite in order to reflect an increase in the performance index for the use of any pursuit control.

4) The R_e matrix is chosen positive definite in order to reflect a decrease in the performance index for the use of any evasion control.

5) The matrices Q, R_p, and R_e are chosen symmetric since this choice involves no loss of generality in the quadratic form but makes subsequent statements about the solution simpler.

3. DERIVATION OF THE OPTIMAL STRATEGY

The derivation given here was first done by Dr. Donald E. Kirk, Assistant Professor of Electrical Engineering at the Naval Postgraduate School. It has been expanded slightly to include a set of sufficiency conditions and to make its application as general as possible.

3.1 Minimaximization of the Performance Index

The optimal controls are found by applying the dynamic programming approach of Bellman [5]. The "cost" of operation during the final (Nth) stage of operation is given by

$$J_1 = [\underline{x}_p(N) - \underline{x}_e(N)]^T Q[\underline{x}_p(N) - \underline{x}_e(N)]$$

$$+ \underline{u}^T(N-1)R_p(N-1)\underline{u}(N-1) - \underline{v}^T(N-1)R_e(N-1)\underline{v}(N-1) . \qquad (15)$$

The effect of the sample period in equation (12) can be accounted for by multiplying each element in the R_p and R_e matrices by T. T is omitted for convenience at this point to be reintroduced later in the derivation. Since $\underline{x}_p(N)$ and $\underline{x}_e(N)$ depend only on $\underline{x}_p(N-1)$, $\underline{x}_e(N-1)$, $\underline{u}(N-1)$, and $\underline{v}(N-1)$, substituting equations (3) and (4) into equation (15) and neglecting the indices, which are all (N-1), yields

$$J_1 = [\Phi_p\underline{x}_p + \Delta_p\underline{u} - \Phi_e\underline{x}_e - \Delta_e\underline{v}]^T Q[\Phi_p\underline{x}_p + \Delta_p\underline{u} - \Phi_e\underline{x}_e - \Delta_e\underline{v}]$$

$$+ \underline{u}^T R_p\underline{u} - \underline{v}^T R_e\underline{v} . \qquad (16)$$

It is convenient to define:

$$\underline{z}(N-1) = \Phi_p\underline{x}_p(N-1) - \Phi_e\underline{x}_e(N-1) . \qquad (17)$$

Note that the $\underline{z}(N-1)$ vector is the projected difference in states at the final time if no control is applied during the final (Nth) stage of operation.

Using equation (17) in (16) and defining P(0) = Q:

$$J_1 = [\underline{z} + \Delta_p\underline{u} - \Delta_e\underline{v}]^T P(0)[\underline{z} + \Delta_p\underline{u} - \Delta_e\underline{v}]$$

$$+ \underline{u}^T R_p\underline{u} - \underline{v}^T R_e\underline{v} . \qquad (18)$$

Defining $dJ_1(d\underline{u})$ and $dJ_1(d\underline{v})$ as the increments of J_1 from its optimal value caused by small variations $d\underline{u}(N-1)$ and $d\underline{v}(N-1)$, respectively, from the optimal values $\underline{u}^o(N-1)$ and $\underline{v}^o(N-1)$ gives:

$$dJ_1(d\underline{u}) = 2\{[\Delta_p^T P(0)\Delta_p + R_p]\underline{u}^o - \Delta_p^T P(0)\Delta_e\underline{v}^o + \Delta_p^T P(0)\underline{z}\}^T d\underline{u}$$

$$+ 2d\underline{u}^T[\Delta_p^T P(0)\Delta_p + R_p]d\underline{u} \qquad (19a)$$

$$dJ_1(d\underline{v}) = 2\{[\Delta_e^T P(0)\Delta_e - R_e]\underline{v}^o - \Delta_e^T P(0)\Delta_p\underline{u}^o - \Delta_e^T P(0)\underline{z}\}^T d\underline{v}$$

$$+ 2d\underline{v}^T[\Delta_e^T P(0)\Delta_e - R_e]d\underline{v} \quad . \qquad (19b)$$

Inspection shows that the necessary conditions for the desired saddle point are that the first order terms in $d\underline{u}$ and $d\underline{v}$ in equations (19) must be zero. Then:

$$[\Delta_p^T P(0)\Delta_p + R_p]\underline{u}^o - \Delta_p^T P(0)\Delta_e\underline{v}^o + \Delta_p^T P(0)\underline{z} = \underline{0} \qquad (20)$$

$$-\Delta_e^T P(0)\Delta_p\underline{u}^o + [\Delta_e^T P(0)\Delta_e - R_e]\underline{v}^o - \Delta_e^T P(0)\underline{z} = \underline{0} \quad . \qquad (21)$$

Assuming that the inverse required below exists, and solving for $\underline{u}^o$ and $\underline{v}^o$ gives:

$$\begin{bmatrix} \underline{u}^o \\ \hline \underline{v}^o \end{bmatrix} = \left[\begin{array}{c|c} \Delta_p^T P(0)\Delta_p + R_p & -\Delta_p^T P(0)\Delta_e \\ \hline -\Delta_e^T P(0)\Delta_p & \Delta_e^T P(0)\Delta_e - R_e \end{array}\right]^{-1} \begin{bmatrix} -\Delta_p^T P(0) \\ \hline \Delta_e^T P(0) \end{bmatrix} \underline{z} \quad . \qquad (22)$$

It is convenient to rewrite equation (22) in the form:

$$\begin{bmatrix} \underline{u}^o(N-1) \\ \hline \underline{v}^o(N-1) \end{bmatrix} = \begin{bmatrix} F_p(1) \\ \hline F_e(1) \end{bmatrix} \underline{z}(N-1) \qquad (23)$$

Substitution of the above expression for the optimal controls $\underline{u}^{o}(N-1)$ and $\underline{v}^{o}(N-1)$ into equation (18) yields the optimal "cost" of operation over the last (Nth) stage of operation:

$$J_1^{o}[\underline{z}(N-1)] = [\underline{z}^T + (\Delta_p F_p \underline{z} - \Delta_e F_e \underline{z})^T] P(0)[\underline{z} + \Delta_p F_p \underline{z} - \Delta_e F_e \underline{z}]$$

$$+ \underline{z}^T F_p^T R_p F_p \underline{z} - \underline{z}^T F_e^T R_e \underline{z} \quad . \tag{24}$$

Now defining $\psi(1) = I + \Delta_p F_p(1) - \Delta_e F_e(1)$, (25)

equation (24) can be rewritten:

$$J_1^{o}[\underline{z}(N-1)] = \underline{z}^T \psi^T(1) P(0) \psi(1) \underline{z} + \underline{z}^T F_p^T(1) R_p(N-1) F_p(1) \underline{z}$$

$$- \underline{z}^T F_e^T(1) R_e(N-1) F_e(1) \underline{z} \quad . \tag{26}$$

Now defining

$$P(i) = \psi^T(i) P(i-1) \psi(i) + F_p^T(i) R_p(N-i) F_p(i) - F_e^T(i) R_e(N-i) F_e(i) \ , \tag{27}$$

equation (26) can be rewritten:

$$J_1^{o}[\underline{z}(N-1)] = \underline{z}^T(N-1) P(1) \underline{z}(N-1) \ . \tag{28}$$

The dynamic programming process is continued by considering the cost of operation over the last two stages of operation.

$$J_2 = \underline{u}^T(N-2) R_p(N-2) \underline{u}(N-2) - \underline{v}^T(N-2) R_e(N-2) \underline{v}(N-2) + J_1[\underline{z}(N-1)] \tag{29}$$

$$J_2^{o} = \min_{\underline{u}(N-1),\ \underline{u}(N-2)} \left[\max_{\underline{v}(N-1),\ \underline{v}(N-2)} J_2 \right] \tag{30}$$

Since $\underline{u}(N-1)$ and $\underline{v}(N-1)$ affect only the J_1 portion of equation (29):

$$J_2^{o} = \min_{\underline{u}(N-2)} \left\{ \max_{\underline{v}(N-2)} \left[\underline{u}^T(N-2) R_p(N-2) \underline{u}(N-2) - \underline{v}^T(N-2) R_e(N-2) \underline{v}(N-2) \right.\right.$$

$$\left.\left. + \min_{\underline{u}(N-1)} \left[\max_{\underline{v}(N-1)} J_1 \right] \right] \right\} \tag{31}$$

Equation (31) can be rewritten using the definition of J_1^o in equation (14):

$$J_2^o = \min_{\underline{u}(N-2)} \left[\max_{\underline{v}(N-2)} \left[\underline{u}^T(N-2)R_p(N-2)\underline{u}(N-2) - \underline{v}^T(N-2)R_e(N-2)\underline{v}(N-2) + J_1^o \right] \right] . \quad (32)$$

Substituting equation (28) into (32) gives:

$$J_2^o = \min_{\underline{u}(N-2)} \left[\max_{\underline{v}(N-2)} \left[\underline{u}^T(N-2)R_p(N-2)\underline{u}(N-2) - \underline{v}^T(N-2)R_e(N-2)\underline{v}(N-2) + \underline{z}^T(N-1)P(1)\underline{z}(N-1) \right] \right] . \quad (33)$$

Relating $\underline{z}(N-1)$ to $\underline{x}_p(N-2)$, $\underline{x}_e(N-2)$, $\underline{u}(N-2)$, and $\underline{v}(N-2)$ yields:

$$\underline{z}(N-1) = \phi_p \underline{x}_p(N-1) - \phi_e \underline{x}_e(N-1) \quad (34)$$

$$\underline{z}(N-1) = \phi_p [\phi_p \underline{x}(N-2) + \Delta_p \underline{u}(N-2)] - \phi_e [\phi_e \underline{x}_e(N-2) + \Delta_e \underline{v}(N-2)] \quad (35)$$

$$\underline{z}(N-1) = \phi_p^2 \underline{x}_p(N-2) - \phi_e^2(N-2) + \phi_p \Delta_p \underline{u}(N-2) - \phi_e \Delta_e \underline{v}(N-2) . \quad (36)$$

Extending the definition of $\underline{z}$, define:

$$\underline{z}(N-i) = \phi_p^i \underline{x}_p(N-i) - \phi_e^i \underline{x}_e(N-i) . \quad (37)$$

It is noted that $\underline{z}(N-i)$ is the projected "miss distance" if no control is applied to either the pursuer or evader after the $(N-i)$th stage of operation.

It is also convenient to define:

$$\Delta_p(i) = \phi_p^i \Delta_p \quad (38)$$

$$\Delta_e(i) = \phi_e^i \Delta_e . \quad (39)$$

Substituting equations (37), (38), and (39) into equation (36) gives:

$$\underline{z}(N-1) = \underline{z}(N-2) + \Delta_p(1)\underline{u}(N-2) - \Delta_e(1)\underline{v}(N-2) \quad . \tag{40}$$

Substituting equation (40) into equation (33), and omitting all (N-2) subscripts gives:

$$J_2^{\,o}[\underline{z}(N-2)] = \min_{\underline{u}(N-2)} \left[\max_{\underline{v}(N-2)} \left\{ \underline{u}^T R_p \underline{u} - \underline{v}^T R_e \underline{v} + [\underline{z} + \Delta_p(1)\underline{u} - \Delta_e(1)\underline{v}]^T P(1) [\underline{z} + \Delta_p(1)\underline{u} - \Delta_e(1)\underline{v}] \right\} \right] \quad . \tag{41}$$

Comparison of equation (41) and equation (18) shows them to be identical in form. It is apparent that the solution shown in equation (23) for the last stage of operation can be generalized for all stages of operation:

$$\begin{bmatrix} \underline{u}^o(N-i) \\ \hline \underline{v}^o(N-i) \end{bmatrix} = \begin{bmatrix} F_p(i) \\ \hline F_e(i) \end{bmatrix} \underline{z}(N-i) = F(i)\underline{z}(N-i) \quad . \tag{42}$$

The recursive relationships necessary to compute the F(i) matrices for all sample periods are as follows:

$$P(0) = Q \tag{43a}$$

$$\Delta_p(0) = \Delta_p \tag{43b}$$

$$\Delta_e(0) = \Delta_e \tag{43c}$$

$$\begin{bmatrix} F_p(i) \\ \hline F_e(i) \end{bmatrix} = \left[\begin{array}{c|c} \Delta_p^T(i-1)P(i-1)\Delta_p(i-1) + R_p(N-i) & -\Delta_p^T(i-1)P(i-1)\Delta_e(i-1) \\ \hline -\Delta_e^T(i-1)P(i-1)\Delta_p(i-1) & \Delta_e^T(i-1)P(i-1)\Delta_e(i-1) - R_e(N-i) \end{array} \right]^{-1} \times \begin{bmatrix} -\Delta_p^T(i-1)P(i-1) \\ \hline \Delta_e^T(i-1)P(i-1) \end{bmatrix} \quad . \tag{43d}$$

$$\psi(i) = [I + \Delta_p(i-1)F_p(i) - \Delta_e(i-1)F_e(i)] \tag{43e}$$

$$P(i) = \psi^T(i)P(i-1)\psi(i) + F_p^T(i)R_p(N-i)F_p(i)$$

$$- F_e^T(i)R_e(N-i)F_e(i) \tag{43f}$$

$$\Delta_p(i) = \phi_p \Delta_p(i-1) \tag{43g}$$

$$\Delta_e(i) = \phi_e \Delta_e(i-1) \quad . \tag{43h}$$

The relationships necessary to generate the optimal controls are:

$$\underline{z}(N-i) = \phi_p^i \underline{x}_p(N-i) - \phi_e^i \underline{x}_e(N-i) \tag{44a}$$

$$\begin{bmatrix} \underline{u}^o(N-i) \\ \hline \underline{v}^o(N-i) \end{bmatrix} = \begin{bmatrix} F_p(i) \\ \hline F_e(i) \end{bmatrix} \underline{z}(N-i) \quad . \tag{44b}$$

The effect of the sample period on the solution, which was neglected after equation (12), can now be readily accounted for by multiplying each element of the R_p and R_e matrices by the sample period.

3.2 Sufficiency Conditions

It has been assumed thus far that the saddle point described in equations (13) and (14) exists. Examination of equations (19) shows that such a saddle point is indeed reached by applying the control law described by equations (43) and (44) if the following conditions are met:

a) the first order terms in $d\underline{u}$ and $d\underline{v}$ in equations (19) must be identically zero,

b) the second order term in $d\underline{u}$ in equation (19a) must be positive definite, and

c) the second order term in $d\underline{v}$ in equation (19b) must be negative definite.

It is convenient here to identify the matrix which requires inversion in equation (43d) as the matrix D, and the sub-matrices into which it is partitioned as follows:

$$D = \left[\begin{array}{c|c} D_{11} & D_{12} \\ \hline D_{21} & D_{22} \end{array}\right] \quad . \tag{45}$$

An equivalent set of sufficient conditions can now be stated as follows:

a) the inverse of the D matrix must exist,

b) the matrix D_{11} must be positive definite, and

c) the matrix D_{22} must be negative definite.

A moment's examination of equation (43d) reveals that the D matrix is symmetric. It can be shown that the inverse of the symmetric matrix D exists if the inverse of D_{11} and the inverse of ξ exist [6], where:

$$\xi = D_{22} - D_{21}(D_{11})^{-1}D_{12} \quad . \tag{46}$$

Therefore, the necessary and sufficient conditions to insure the required saddle point are:

a) $\Delta_p^T(i-1)P(i-1)\Delta_p(i-1) + R_p(N-i)$ must be positive definite for all i from one to N, and

b) $\Delta_e^T(i-1)P(i-1)\Delta_e(i-1) - R_e(N-i)$ must be negative definite for all i from one to N.

4. DIGITAL COMPUTER SIMULATION PROGRAM

In order to simulate the results of applying the control strategy derived in this thesis to various systems, the digital computer program OPTIMAL2 was employed. This program was written in FORTRAN 63 computer language for use with a Control Data Corporation 1604 Computer. The complete program is given in Appendix A.

The program OPTIMAL2 was written to have as general an application as possible within the restrictions imposed in the problem statement and derivation. To this end, pursuit and evasion system descriptions and the performance index to be minimized are read in as data. (The pursuit and evasion systems are completely described by the differential equation coefficient matrices A_p, A_e, B_p, B_e, and the sample period T. The performance index to be minimized is completely described by the weighting matrices Q, R_p, R_e, the sample period T, and the number of samples N. The initial conditions $\underline{x}_p(0)$ and $\underline{x}_e(0)$ are required for trajectory simulation only.) The program will accept any system so described with state vector dimensions of ten or less and with control vectors of dimensions such that the sum for both plants does not exceed ten. Printout of input data is provided for verification purposes.

The program uses subroutine PHIDEL to compute the matrices $\emptyset_p$, $\emptyset_e$, Δ_p, and Δ_e for the systems described by the input data. This computation is effected by programming truncated forms of the infinite series given in equations (7) and (10). Printout of these matrices is provided for reference purposes.

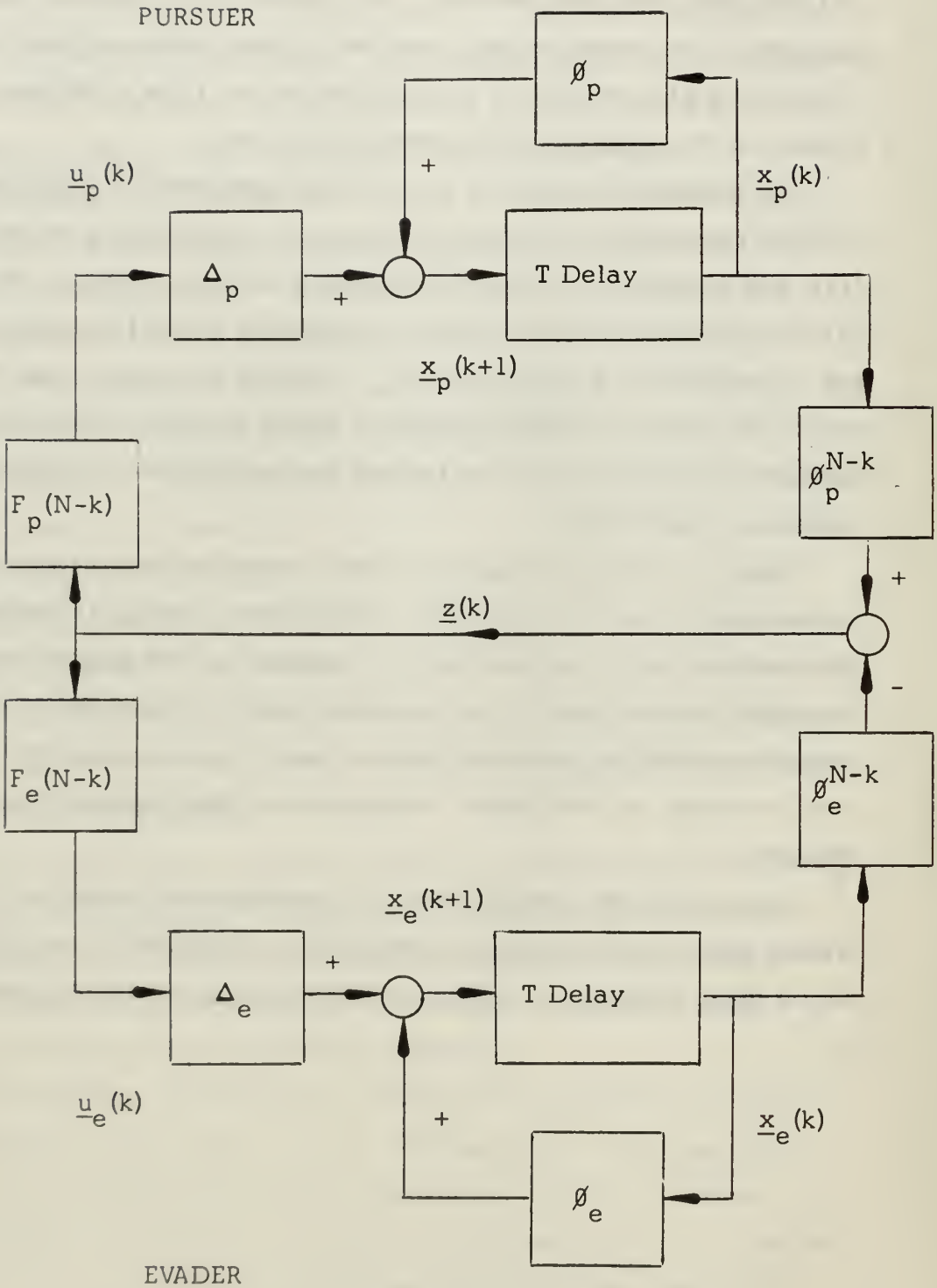
The program uses subroutine OPTCON2 to generate and record on magnetic tape for later use the "feedback gain" matrices $F_p(i)$ and $F_e(i)$ and the state transition matrices $\emptyset_p^i$ and $\emptyset_e^i$. These computations are effected by programming equations (43). Comment statements in the subroutine indicate the appropriate location for additional statements required to read from data or to generate a "time-varying" form of the control weighting matrices $R_p(N-i)$ and

$R_e(N-i)$. (No correction is required if these matrices are constant.) Provision is made for printout of a number of the feedback-gain matrices and the state transition matrices, and for graphical output of the (1,1) elements of the feedback-gain matrices to show trends in their variation. Provision is also made for a printout indication if the sufficiency conditions for the required saddle point are not met.

The program then applies the matrices generated by OPTCON2 to compute the sequence of optimal controls by programming equations (44), and generates a simulated trajectory for each system. The value of the performance index is computed for these trajectories, and is provided as a printed output. Comment statements near the end of the program OPTIMAL2 indicate where changes of the output statements may be required to present the trajectories of different systems in usable form.

Figure 1 shows a vector flow-graph model for implementing the optimal pursuit-evasion strategy. The optimal strategy is defined (the feedback-gain matrices may be computed for the largest number of sample periods likely to be required) when the dynamics of the pursuit and evasion plants are known, and it may be recorded. Therefore, "on-line" or "real-time" computation of the strategy is not required.

Examples of the required form of input data and examples of printed output are presented in Appendix B. Examples of variations on the basic program for different systems are presented in Appendix C.



VECTOR FLOWGRAPH MODEL

FIGURE 1

5. RESULTS OF COMPUTER SIMULATION TESTS

Using the computer simulation program described in the preceding section, three series of tests were run to investigate different characteristics of the optimal pursuit-evasion strategy. The results are reported below.

The following definitions are used to simplify the recording of data:

$$\underline{z}(N) = \underline{x}_p(N) - \underline{x}_e(N) , \tag{47}$$

$$\text{Cost R} = \sum_{k=0}^{N} [\underline{u}^T(k)R_p\underline{u}(k) - \underline{v}^T(k)R_e\underline{v}(k)] , \tag{48}$$

$$\text{Cost Q} = \underline{z}^T(N)Q\underline{z}(N) , \text{ and} \tag{49}$$

$$J = \text{Cost R} + \text{Cost Q} . \tag{50}$$

5.1 The Effect of Varying the Performance Index Parameters

Run 1 is a standard intercept to which subsequent runs are generally compared. The pursuit and evasion plants are identical, undamped, spherical, unit masses traveling in a two-dimensional (north-east) space. The state vectors and control vectors for these plants are defined:

$$\underline{x} = \begin{bmatrix} \text{north} \\ \dot{\text{north}} \\ \text{east} \\ \dot{\text{east}} \end{bmatrix} \quad \text{and} \quad \underline{u} = \begin{bmatrix} \dot{\text{north}} \\ \dot{\text{east}} \end{bmatrix} , \tag{51}$$

The computer input data and terminal performance data are given below. The resulting trajectories and the time variation of the (1,1) elements of the feedback-gain matrices are shown in Figures 2 and 3. A complete set of input data showing the required format and a complete computer printout of the numerical results for this run are given in Appendix B.

Input Data

$$A_p = A_e = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \qquad B_p = B_e = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix},$$

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \qquad R_p = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad R_e = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix},$$

$$\underline{x}_p(0) = \begin{bmatrix} 0 \\ 5 \\ 0 \\ 5 \end{bmatrix}, \qquad \underline{x}_e(0) = \begin{bmatrix} 500 \\ -10 \\ 0 \\ 10 \end{bmatrix}, \qquad N = 20, \quad T = 1.$$

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} -.0938 \\ 30.0020 \\ -.0469 \\ 2.5012 \end{bmatrix}$$

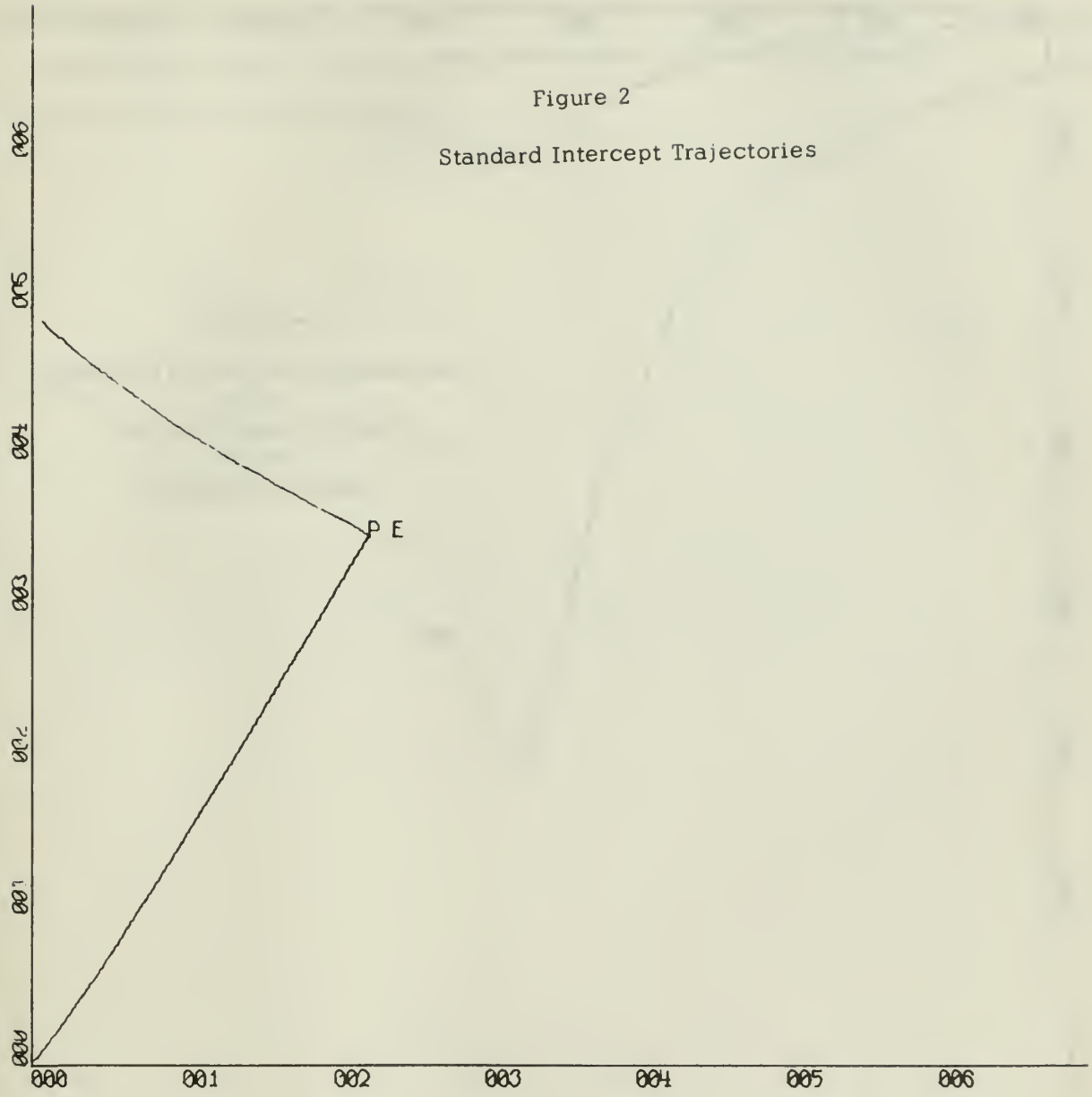
Cost R = 23.430
Cost Q = .011
J = 23.441

The use of this particular semi-definite Q matrix defines position only as the "states-of-interest" for the performance index. The effect of this choice is readily apparent in the resulting relative difference in the magnitudes of the velocity and position elements of the state difference vector at the final time, $\underline{z}(N)$, and in the "intercept" trajectories in Figure 2.

Examination of the series of feedback-gain matrices (see Appendix B), which define the optimal strategy for this run, shows that the required acceleration for the pursuer is always parallel to the projected terminal "miss-distance" vector, so as to provide maximum reduction in the projected miss-distance for a given acceleration magnitude. The required acceleration for the evader is in the same direction (and smaller in magnitude). This situation is peculiar to the special case where both systems are point masses, the controls

Figure 2

Standard Intercept Trajectories



X-SCALE = 1.00E+02 UNITS/INCH.
Y-SCALE = 1.00E+02 UNITS/INCH.

NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 1 2 1 67

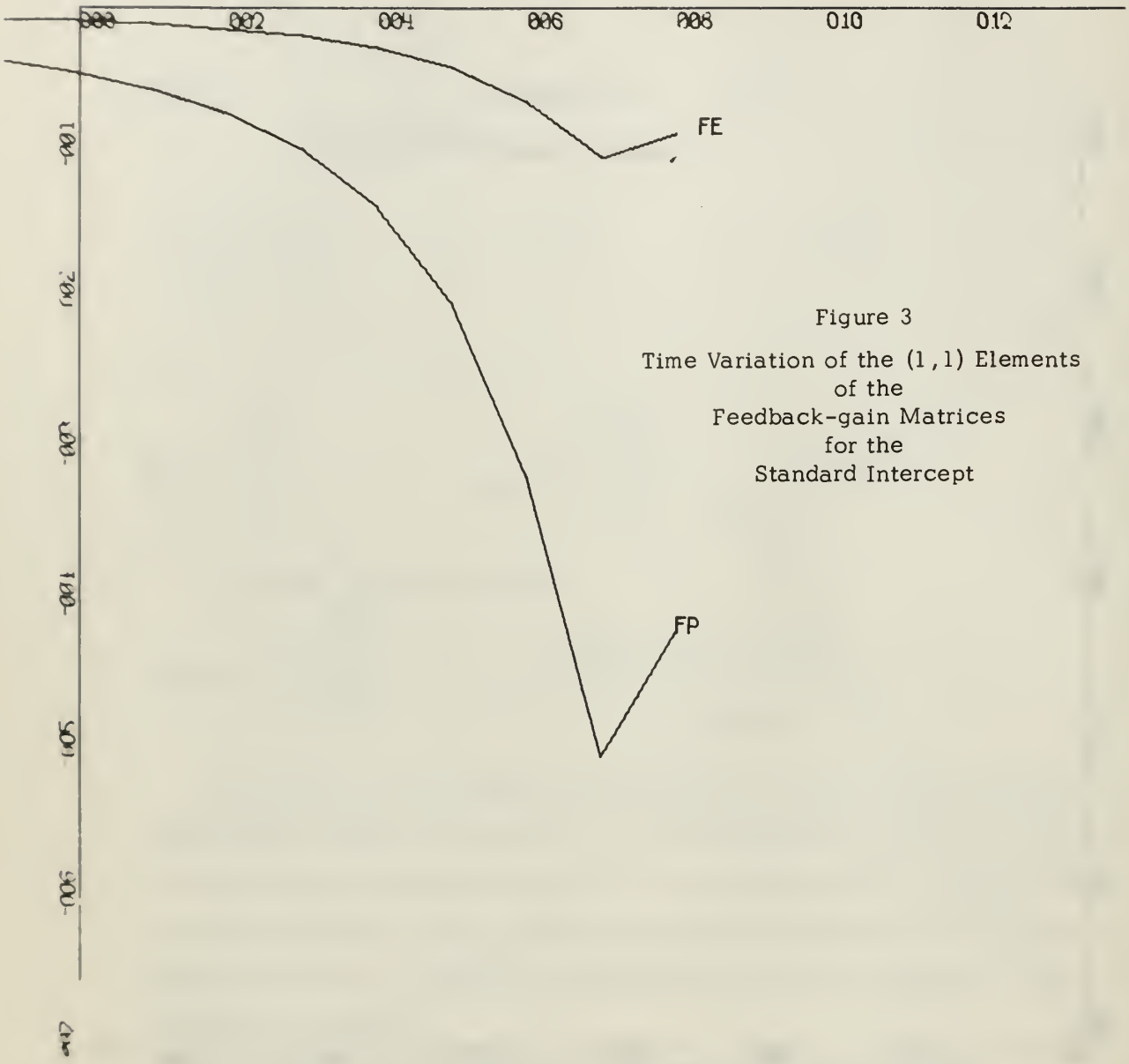


Figure 3

Time Variation of the (1,1) Elements
of the
Feedback-gain Matrices
for the
Standard Intercept

X-SCALE = 2.00E+00 UNITS/INCH.
Y-SCALE = 1.00E-01 UNITS/INCH.

ADD +2.00E+00 UNITS TO ALL X VALUES.

FP(1,1) AND FE(1,1) VS TIME
HUTCHISON,PERMENTER,628,OPTML.2 RUN 1 2 1 67

are accelerations, and the final state weighting matrix Q indicates interest only in position error at the final time. This physical interpretation of the strategy, previously stated only in mathematical form, lends an intuitive appeal.

Run 2 shows the effect of increasing the time for pursuit. The input data and the terminal performance data are given below. The resulting trajectories are shown in Figure 4.

Input Data

Identical to Run 1 except: N = 25.

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} -.030 \\ 22.501 \\ -.030 \\ 2.500 \end{bmatrix}$$

Cost R = 7.4994
Cost Q = .0018

J = 7.5012

The resulting trajectories differ from the standard of Run 1 in that:

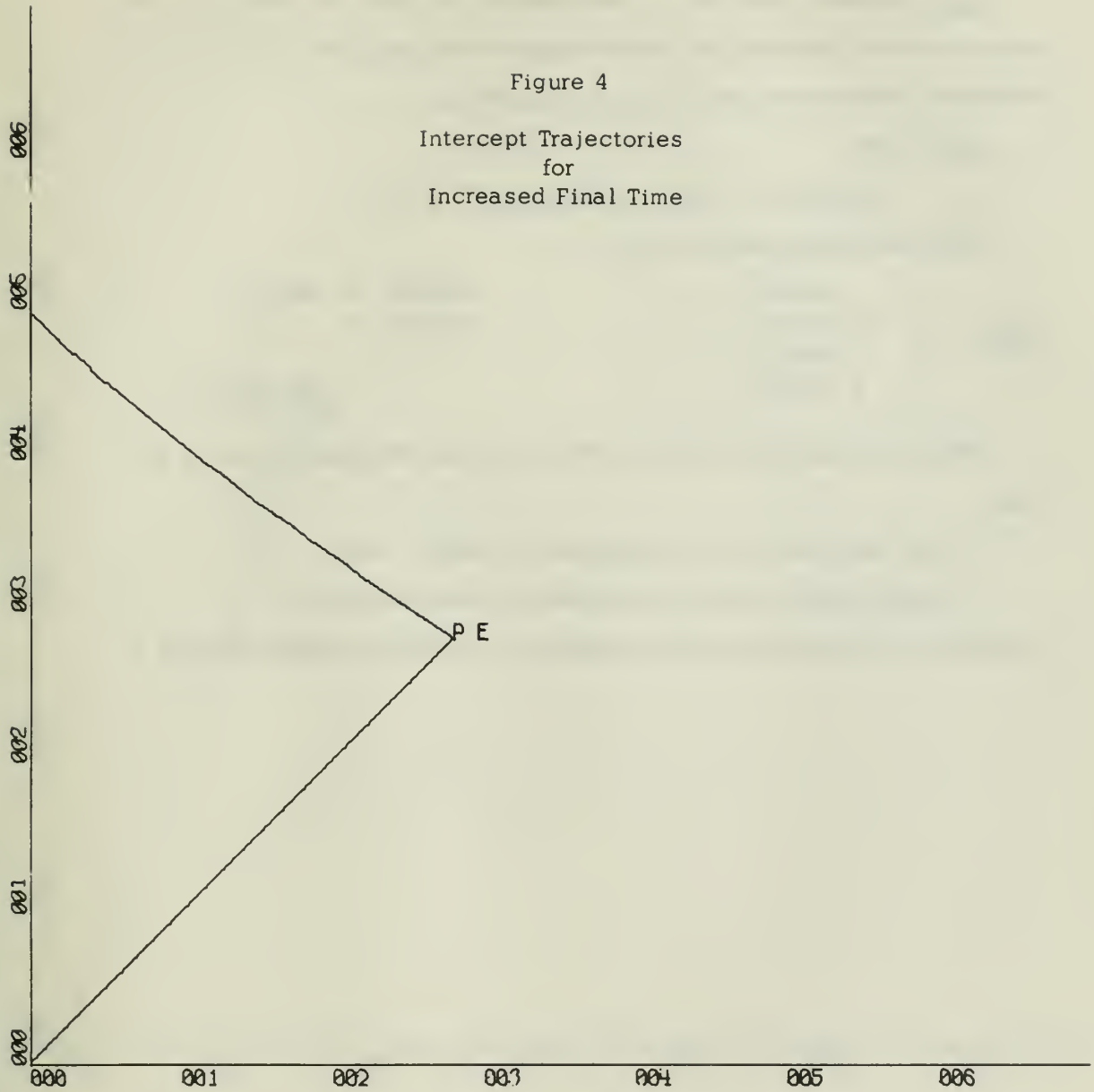
a) the terminal "miss-distance" is less, and

b) smaller magnitudes of acceleration are required.

The series of feedback-gain matrices is identical to that for Run 1.

Figure 4

Intercept Trajectories
for
Increased Final Time



X-SCALE = 1.00E+02 UNITS/INCH.
Y-SCALE = 1.00E+02 UNITS/INCH.
NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 2 2 1 67

Run 3 shows the effect of decreasing the time for pursuit. The input data and the terminal performance data are given below. The resulting trajectories are shown in Figure 5.

Input Data

Identical to Run 1 except: N = 15.

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} -.0305 \\ 42.5010 \\ -.0833 \\ 2.5000 \end{bmatrix}$$

Cost R = 90.117
Cost Q = .101

J = 90.278

The resulting trajectories differ from the standard of Run 1 in that:

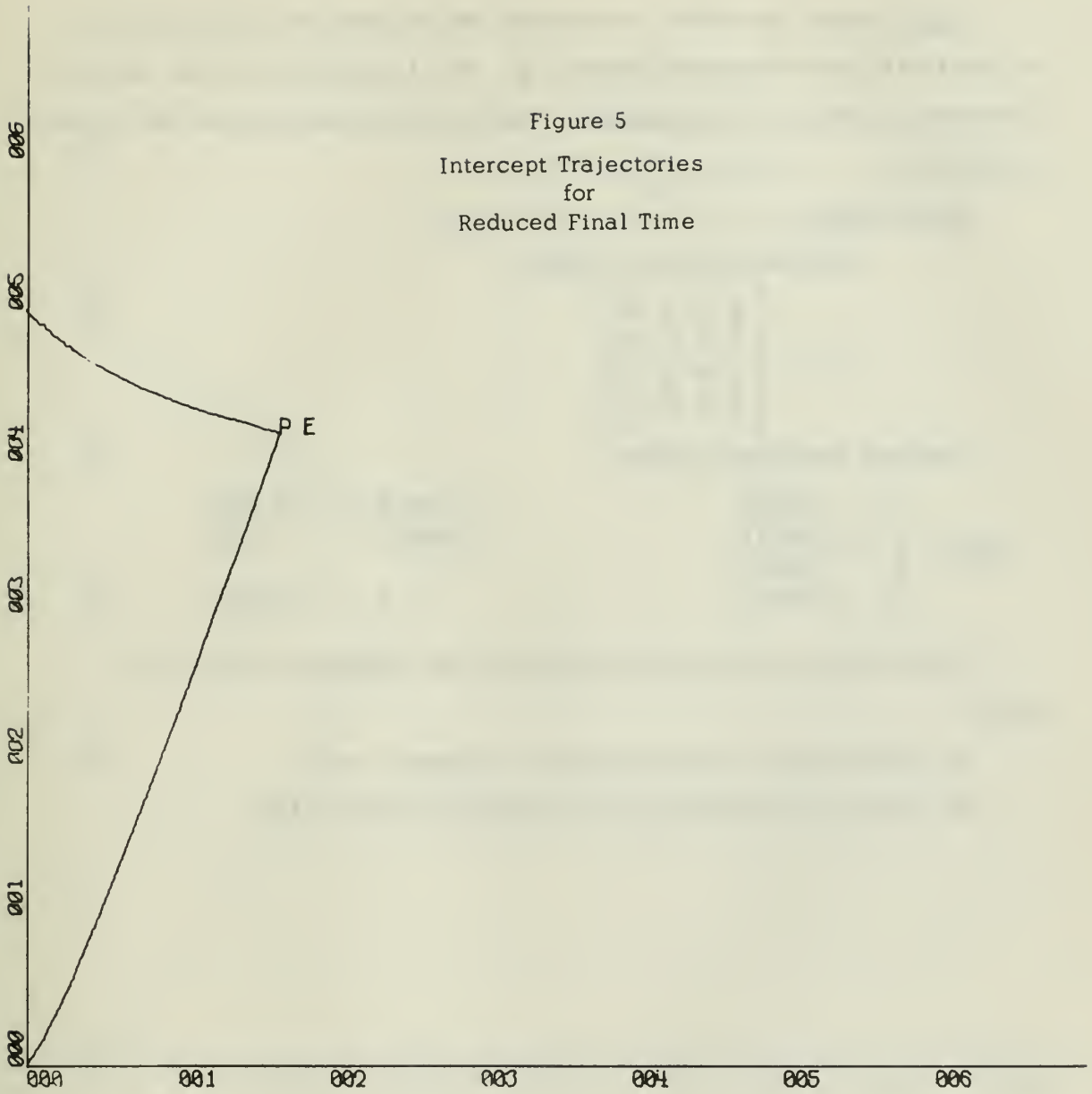
a) the terminal "miss-distance" is larger, and

b) larger magnitudes of acceleration are required.

The series of feedback-gain matrices is identical to that for Run 1.

Figure 5

Intercept Trajectories
for
Reduced Final Time



X-SCALE = 1.00E+02 UNITS/INCH.
Y-SCALE = 1.00E+02 UNITS/INCH.

NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 3 2 1 67

Run 4 shows the effect of reducing the magnitude of the elements of the final-state weighting matrix, Q. The input data and the terminal performance data are given below. The resulting trajectories are shown in Figure 6.

Input Data

Identical to Run 1 except:

$$Q = \begin{bmatrix} \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} -.1874 \\ 29.9950 \\ -.0937 \\ 2.4977 \end{bmatrix}$$

Cost R = 23.408
Cost Q = .022

J = 23.430

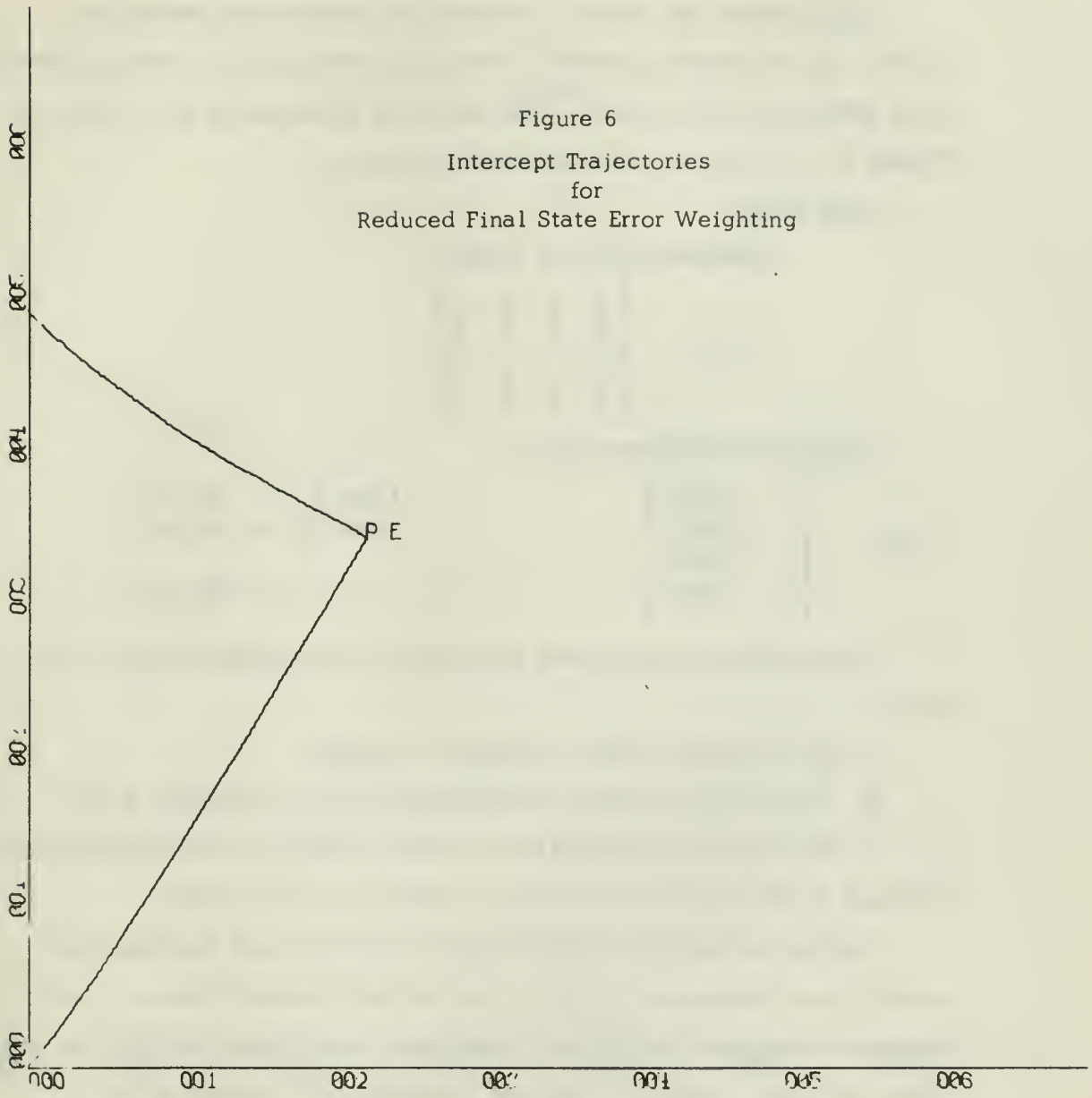
The resulting trajectories differ from the standard of Run 1 in that:

a) the terminal "miss-distance" is larger, and

b) the required acceleration magnitudes are smaller.

Figure 6

Intercept Trajectories
for
Reduced Final State Error Weighting



X-SCALE = 1.00E+02 UNITS/INCH.
Y-SCALE = 1.00E+02 UNITS/INCH.
NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 4 2 1 67

Run 5 shows the effect of making the final-state weighting matrix, Q, an identity matrix. The input data and the terminal performance data are given below. The resulting trajectories are shown in Figure 7.

Input Data

Identical to Run 1 except:

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} -.5440 \\ 6.0027 \\ -.0844 \\ .5004 \end{bmatrix}$$

Cost R = 168.20
Cost Q = 36.59

J = 204.79

The resulting trajectories differ from the standard of Run 1 in that:

a) the terminal "miss-distance" is larger,

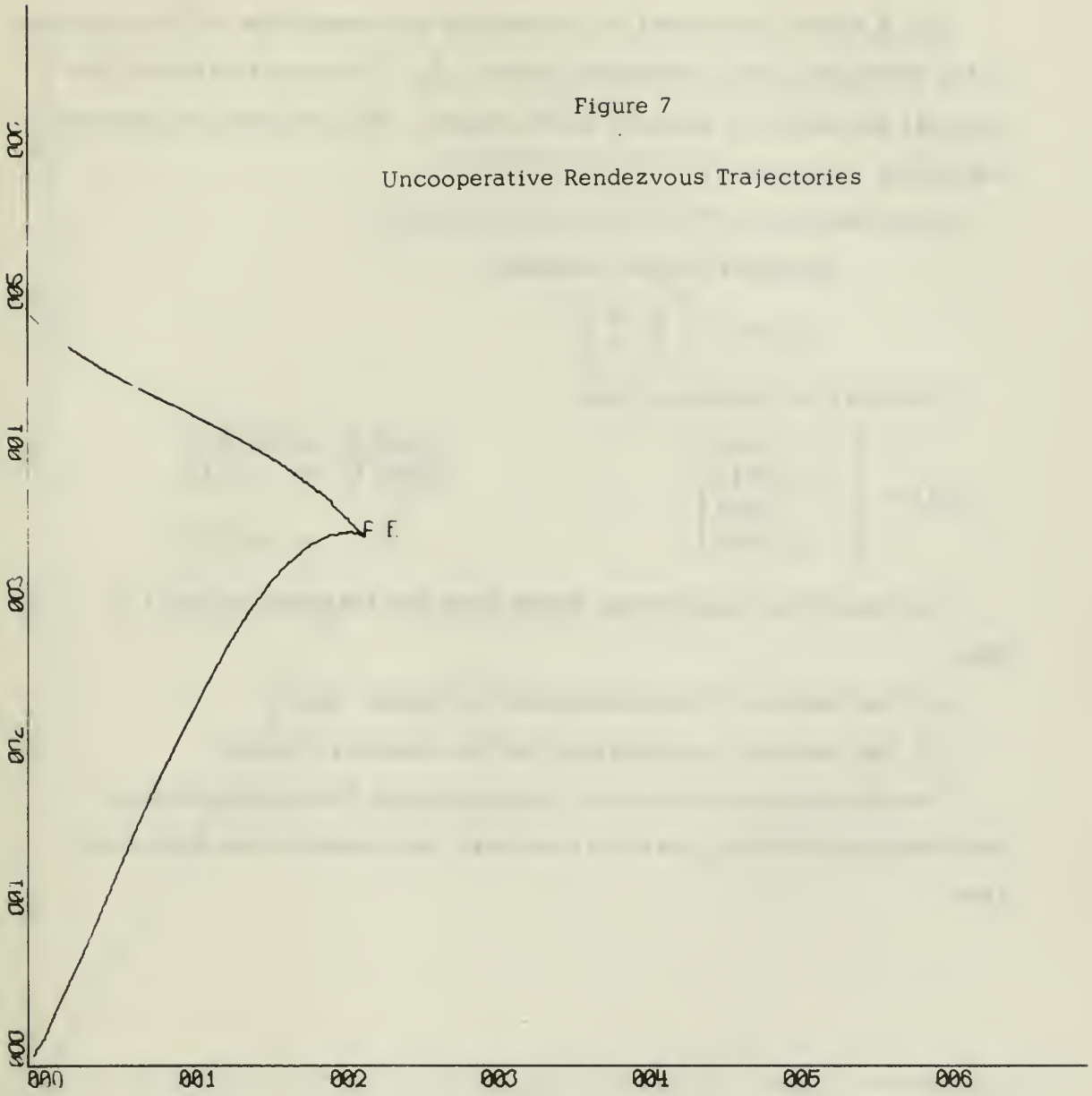
b) the terminal velocity difference vector is smaller, and

c) the required acceleration for this strategy is no longer always parallel to the projected terminal "miss-distance" vector.

The use of this particular Q matrix defines both position and velocity as "states-of-interest" for the performance index. Comparing the terminal performance data and trajectories for this run with those for Run 1 shows a resulting "rendezvous" instead of an "intercept".

Figure 7

Uncooperative Rendezvous Trajectories



X-SCALE = 1.00E+02 UNITS/INCH.
Y-SCALE = 1.00E+02 UNITS/INCH.

NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 5 2 1 67

Run 6 shows the effect of decreasing the magnitude of the elements of the evasion control weighting matrix, R_e. The input data and the terminal performance data are given below. The resulting trajectories are shown in Figure 8.

Input Data

Identical to Run 1 except:

$$R_e = \begin{bmatrix} 3 & 0 \\ 0 & 3 \end{bmatrix}$$

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} -.1125 \\ 30.0010 \\ -.0563 \\ 2.5005 \end{bmatrix}$$

Cost R = 28.111
Cost Q = .016

J = 28.127

The resulting trajectories differ from the standard of Run 1 in that:

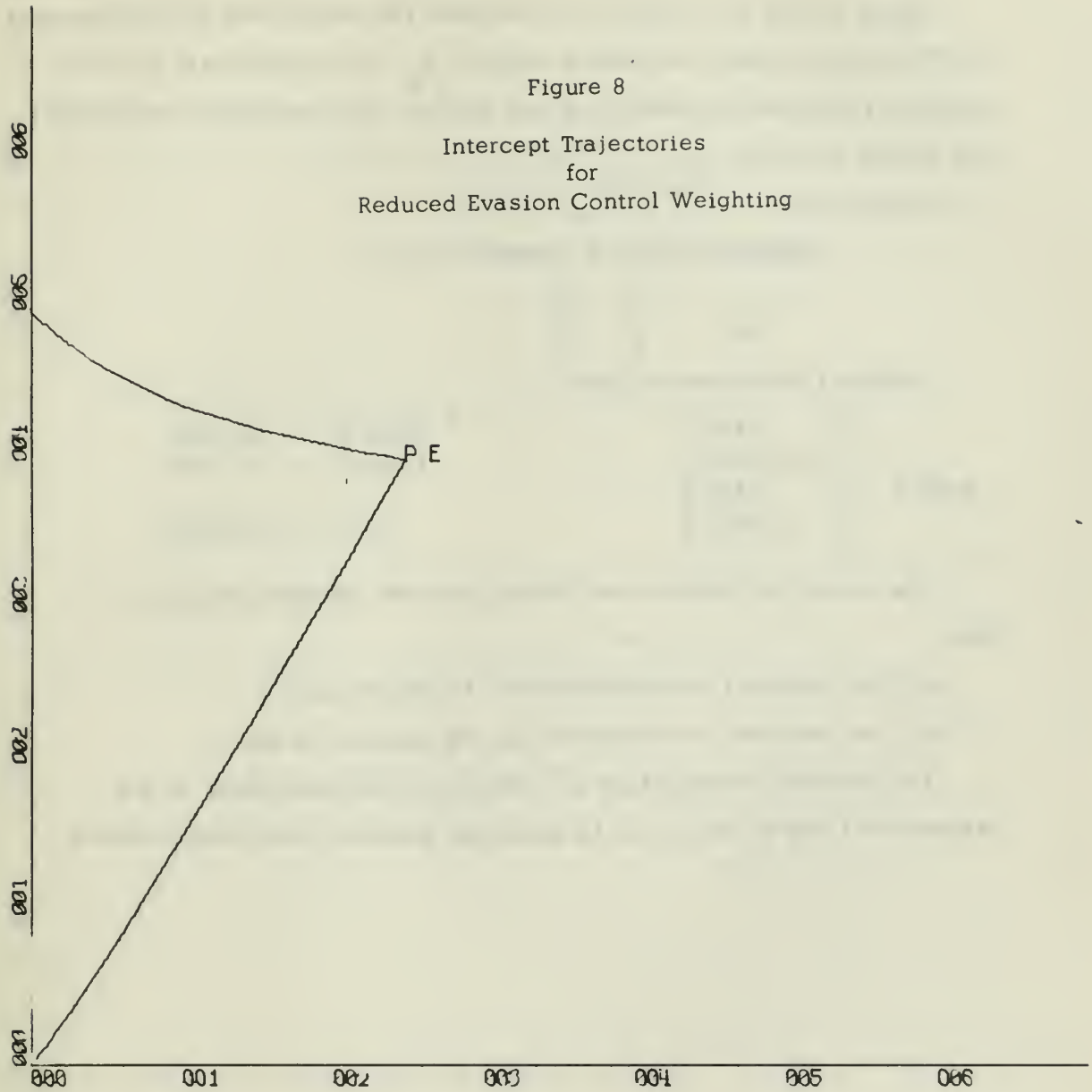
a) the terminal "miss-distance" is larger, and

b) the required acceleration for the evader is larger.

The physical implication of this reduction in the magnitude of the elements of the R_e matrix is to make the evader more maneuverable.

Figure 8

Intercept Trajectories
for
Reduced Evasion Control Weighting



X-SCALE = 1.00E+02 UNITS/INCH.
Y-SCALE = 1.00E+02 UNITS/INCH.
NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 6 2 1 67

Run 7 shows the effect of increasing the magnitude of the elements of the pursuit control weighting matrix, R_p. The input data and the terminal performance data are given below. The resulting trajectories are shown in Figure 9.

Input Data

Identical to Run 1 except:

$$R_p = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} -.2498 \\ 29.9910 \\ -.0469 \\ 2.4977 \end{bmatrix}$$

Cost R = 62.383
Cost Q = .078

J = 62.461

The resulting trajectories differ from the standard of Run 1 in that:

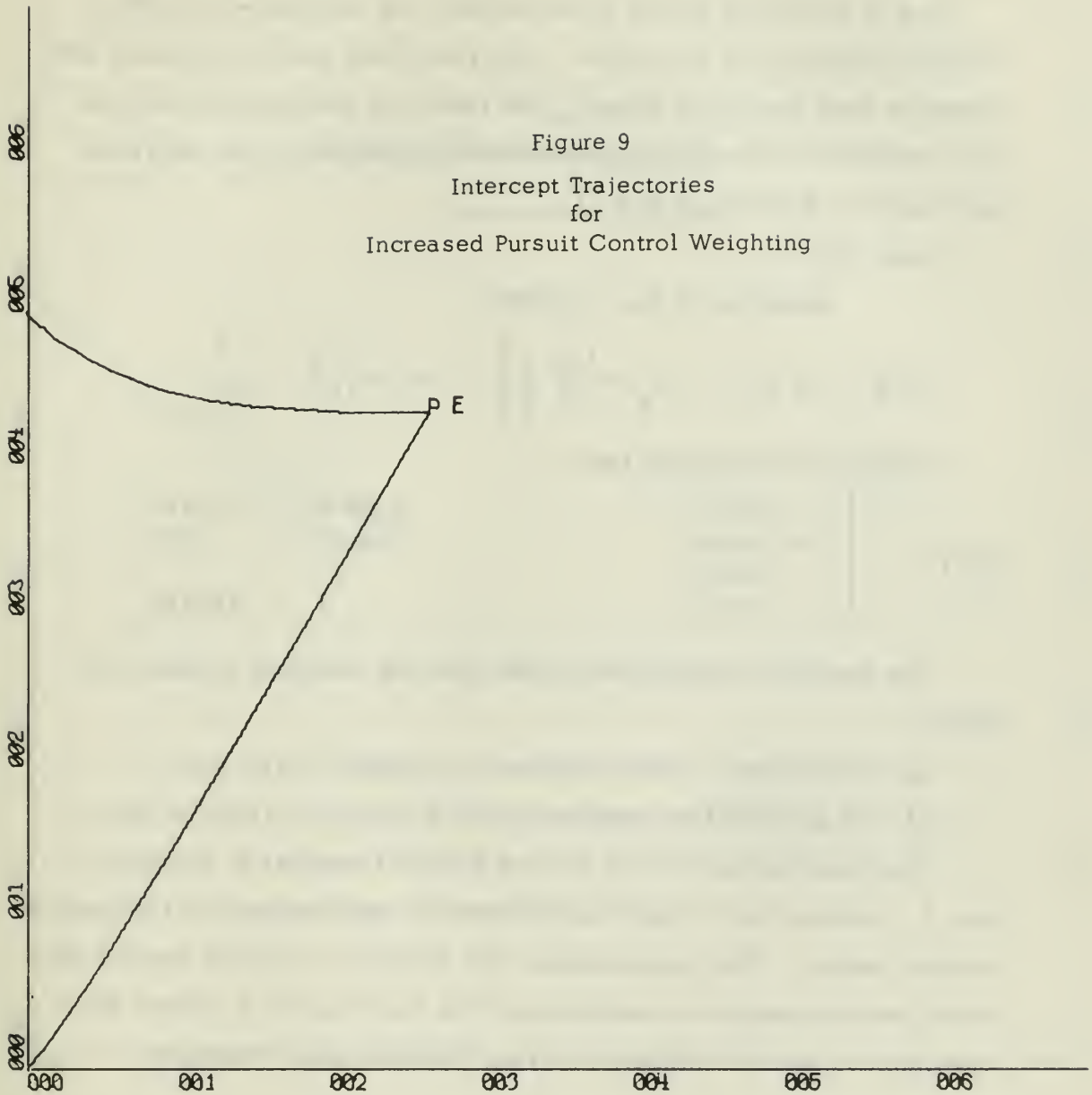
a) the terminal "miss-distance" is larger, and

b) the required acceleration for the pursuer is less.

The physical implication of increasing the magnitude of the elements of the R_p matrix is to make the pursuer less maneuverable.

Figure 9

Intercept Trajectories
for
Increased Pursuit Control Weighting



X-SCALE = 1.00E+02 UNITS/INCH.
Y-SCALE = 1.00E+02 UNITS/INCH.

NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 7 2 1 67

Run 8 shows the effect of decreasing the sampled-data period without changing the final time. The input data and the terminal performance data are given below. The resulting trajectories and the time variation of the (1,1) elements of the feedback-gain matrices are shown in Figures 10 and 11.

Input Data

Identical to Run 1 except:

$$T = \tfrac{1}{2}, \quad N = 40, \quad R_p = \begin{bmatrix} \tfrac{1}{2} & 0 \\ 0 & \tfrac{1}{2} \end{bmatrix}, \quad R_e = \begin{bmatrix} 2\tfrac{1}{2} & 0 \\ 0 & 2\tfrac{1}{2} \end{bmatrix}.$$

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} -.0937 \\ 29.9950 \\ -.0469 \\ 2.4997 \end{bmatrix}$$

Cost R = 23.419
Cost Q = .016

J = 23.430

The resulting trajectories differ from the standard of Run 1 in that:

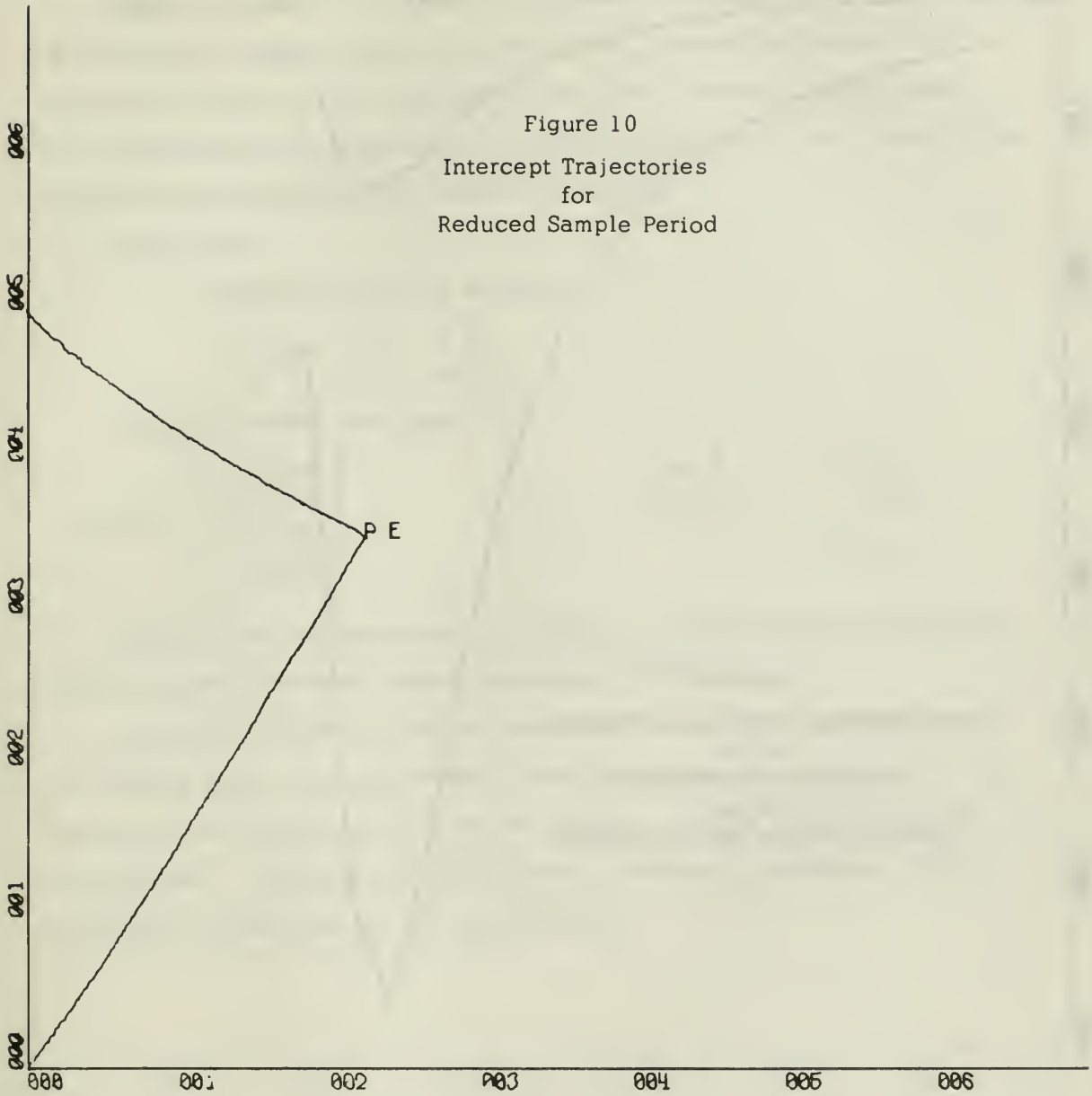
a) the terminal "miss-distance" is slightly less, and

b) the acceleration required for both plants is slightly less.

The trajectories for this run are almost identical to those for Run 1, showing only slight improvement in performance for the smaller sample period. This demonstrates the ability to use this method with small sample periods to approximate the solution for a system with continuous controls and time varying feedback-gain matrices.

Figure 10

Intercept Trajectories
for
Reduced Sample Period



X-SCALE = 1.00E+02 UNITS/INCH.
Y-SCALE = 1.00E+02 UNITS/INCH.
NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 8 2 1 67

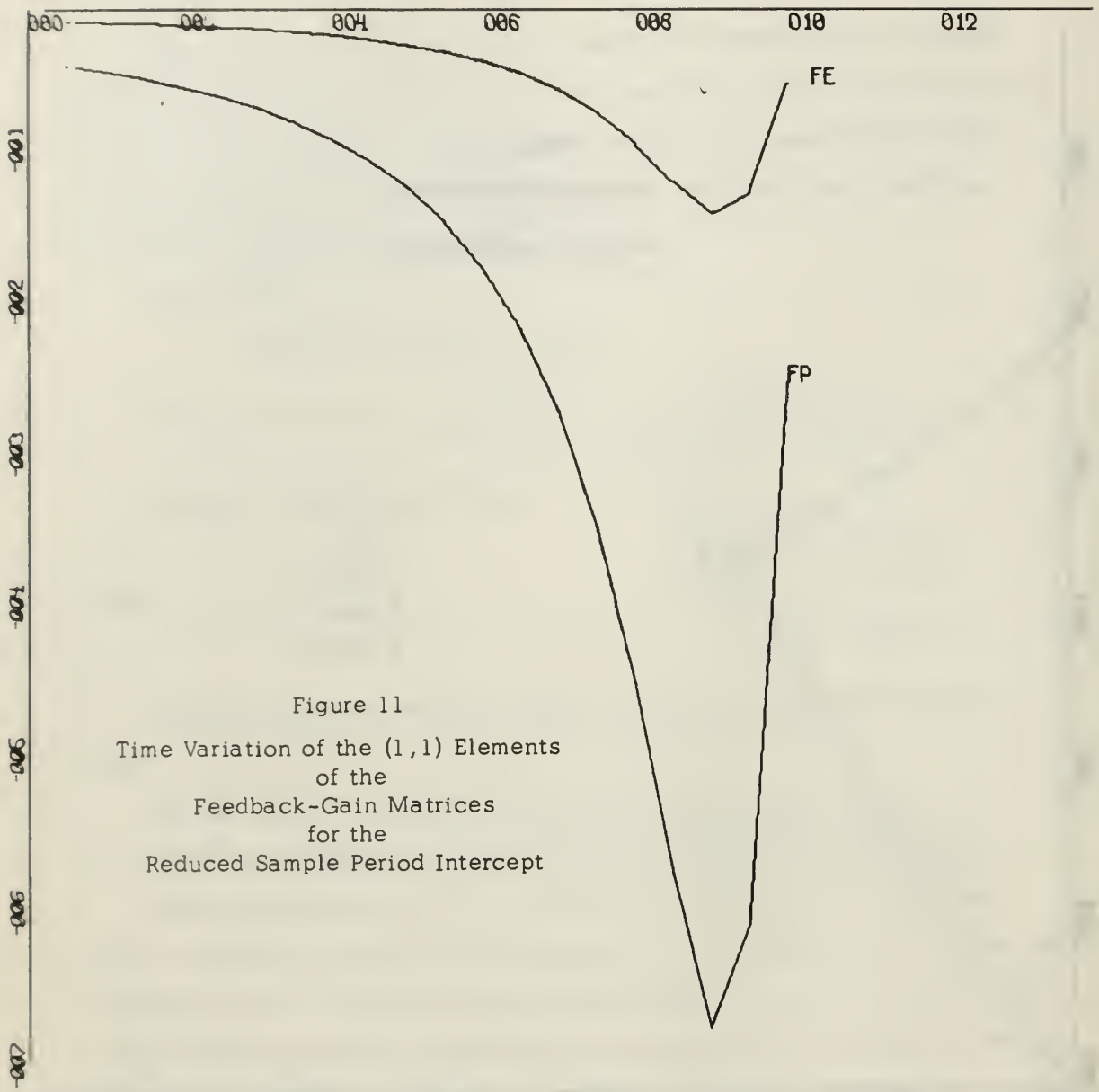


Figure 11

Time Variation of the (1,1) Elements
of the
Feedback-Gain Matrices
for the
Reduced Sample Period Intercept

X-SCALE = 2.00E+00 UNITS/INCH.
Y-SCALE = 1.00E-01 UNITS/INCH.
FP(1,1) AND FE(1,1) VS TIME
HUTCHISON,PERMENTER,628,OPTML.2 RUN 8 2 1 67

Run 9 shows the effect of reducing the magnitude of the elements of the evasion control weighting matrix, R_e, enough to violate the necessary conditions for the existence of the required saddle point. The input data and the terminal performance data are given below. The resulting trajectories are shown in Figure 12.

Input Data

Identical to Run 1 except:

$$R_e = \begin{bmatrix} .4 & 0 \\ 0 & .4 \end{bmatrix}$$

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} .0500 \\ 30.0132 \\ .0250 \\ 2.5066 \end{bmatrix}$$

Cost R = -12.514
Cost Q = .003

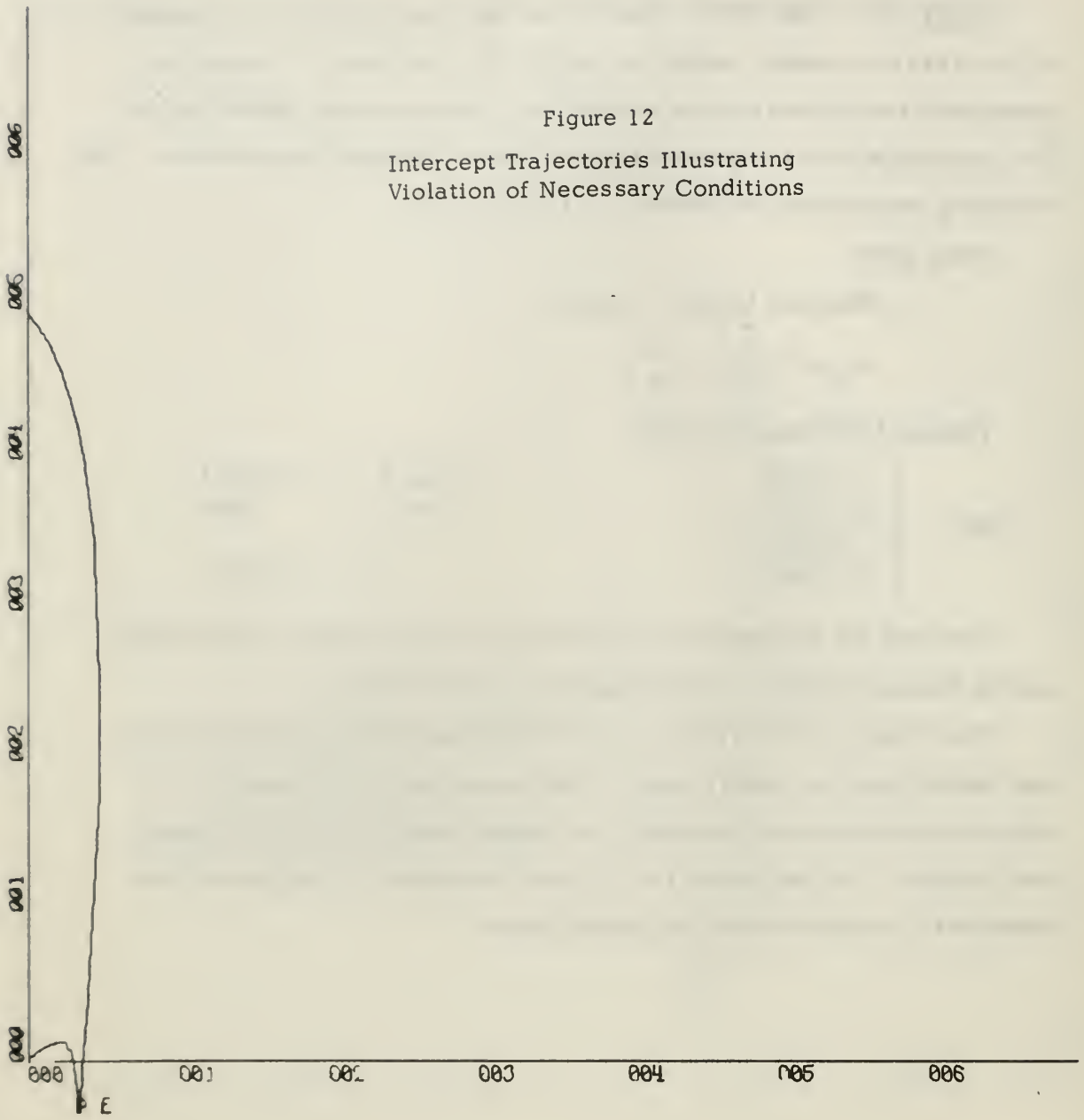
J = -12.511

The test for the necessary conditions for the saddle point failed on the second iteration within subroutine OPTCON2.

The effect of the failure to satisfy the necessary conditions for the saddle point is easily seen in the trajectories of Figure 12. In this case the pursuer runs from the evader, and the evader pursues the pursuer. Further study is indicated regarding violations of the necessary conditions for the saddle point.

Figure 12

Intercept Trajectories Illustrating
Violation of Necessary Conditions



X-SCALE = 1.00E+02 UNITS/INCH
Y-SCALE = 1.00E+02 UNITS/INCH
NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTIML.2 RUN 9 2 1 67

Run 10 is made to determine the point at which decreasing the magnitude of the elements of the evasion control weighting matrix, R_e, causes failure of the test for the required saddle point. No trajectories are simulated. The strategy was computed for input data identical to that for Run 1 except for the values of the R_e matrix which are given below. The existence of the saddle point was tested for 100 sample periods, and the results are as follows:

$$R_e = \begin{bmatrix} 2.7 & 0 \\ 0 & 2.7 \end{bmatrix}, \qquad \text{saddle point exists for } N \leq 100;$$

$$R_e = \begin{bmatrix} 2.6 & 0 \\ 0 & 2.6 \end{bmatrix}, \qquad \text{saddle point exists for } N \leq 100;$$

$$R_e = \begin{bmatrix} 2.5 & 0 \\ 0 & 2.5 \end{bmatrix}, \qquad \text{saddle point does not exist for } N \geq 3;$$

$$R_e = \begin{bmatrix} 0.4 & 0 \\ 0 & 0.4 \end{bmatrix}, \qquad \text{saddle point does not exist for } N \geq 2.$$

This run shows that the saddle point exists for the plants specified for this series of tests, with Q and R_p as specified, for the elements of the R_e matrix greater than 2.5.

5.2 Comparisons of Optimal and Sub-optimal Strategies

Run 11 shows the effect of the evader abandoning the optimal strategy by not maneuvering at all. The input data is identical to that for Run 1. A statement was added in the program to require:

$$\underline{v} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \text{for trajectory simulation.}$$

The pursuer followed the optimal strategy (assuming the evader would do likewise). The resulting trajectories are shown in Figure 13.

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} -.0048 \\ 28.5051 \\ -.0024 \\ 1.7527 \end{bmatrix}$$

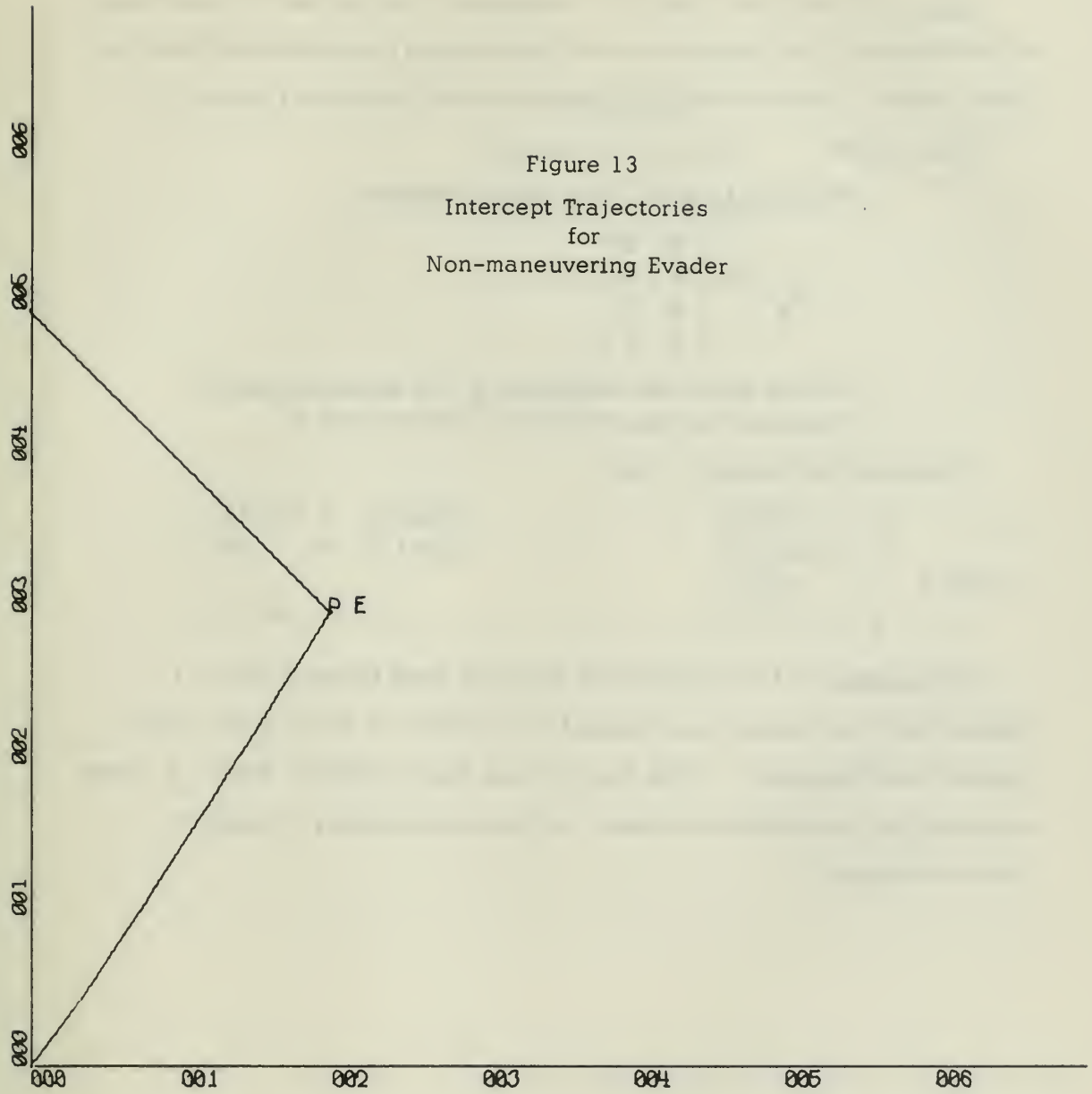
Cost R = 19.708
Cost Q = .000

J = 19.708

Comparison of the results of this test with those of Run 1 shows that (at least for this case) the left-hand inequality of equation (13) does indeed hold. Intuitively, the pursuer does "better" if the evader does not use his "intelligence". The series of feedback-gain matrices is identical to that for Run 1.

Figure 13

Intercept Trajectories
for
Non-maneuvering Evader



X-SCALE = 1.00E+02 UNITS/INCH
Y-SCALE = 1.00E+02 UNITS/INCH
NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 11 2 1 67

Run 12 shows the effect of "improving" the pursuer's knowledge of the evader. The input data and the terminal performance data are given below. The resulting trajectories are shown in Figure 14.

Input Data

Identical to Run 1 and Run 11 except:

$$B_e = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} .$$

(In this case the statement $\underline{v} = 0$ is not required to prevent the evader from maneuvering.)

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} -.0750 \\ 30.0041 \\ -.0375 \\ 2.5019 \end{bmatrix}$$

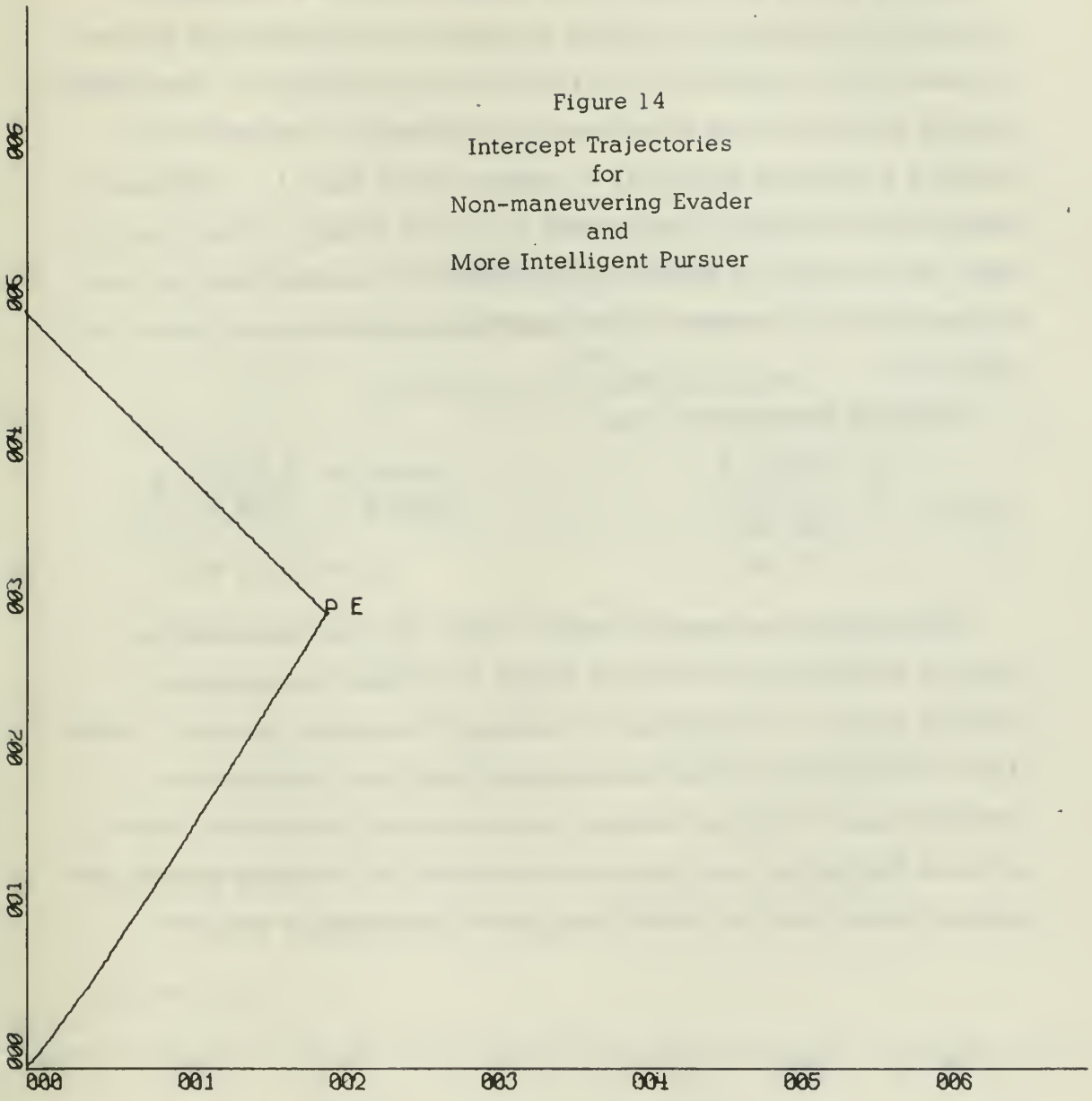
Cost R = 18.748
Cost Q = .007

J = 18.755

Comparison of the results of this run with those of Run 11 shows that the pursuer can indeed do "better" if he has (or uses) "better intelligence". Note that in this case "better" means a lower value of the performance index, and not necessarily a smaller "miss-distance".

Figure 14

Intercept Trajectories
for
Non-maneuvering Evader
and
More Intelligent Pursuer



X-SCALE = 1.00E+02 UNITS/INCH.
Y-SCALE = 1.00E+02 UNITS/INCH.
NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 12 2 1 67

Run 13 shows the effect of the pursuer using the strategy of "proportional navigation" against the same non-maneuvering evader. The input data is identical with that for Run 1 and Run 11. The evader is again prevented from maneuvering by inserting a statement requiring $\underline{v} = \underline{0}$ for the trajectory simulation as in Run 11. Additional statements are added to require the pursuer's speed to remain constant, and his rate of turn to be five times the evaders bearing rate (as seen from the pursuer). The resulting trajectories are shown in Figure 15.

Terminal Performance Data

$$\underline{z}(N) = \begin{bmatrix} -269.53 \\ 6.91 \\ -68.31 \\ -3.34 \end{bmatrix}$$

Cost R = 3.975

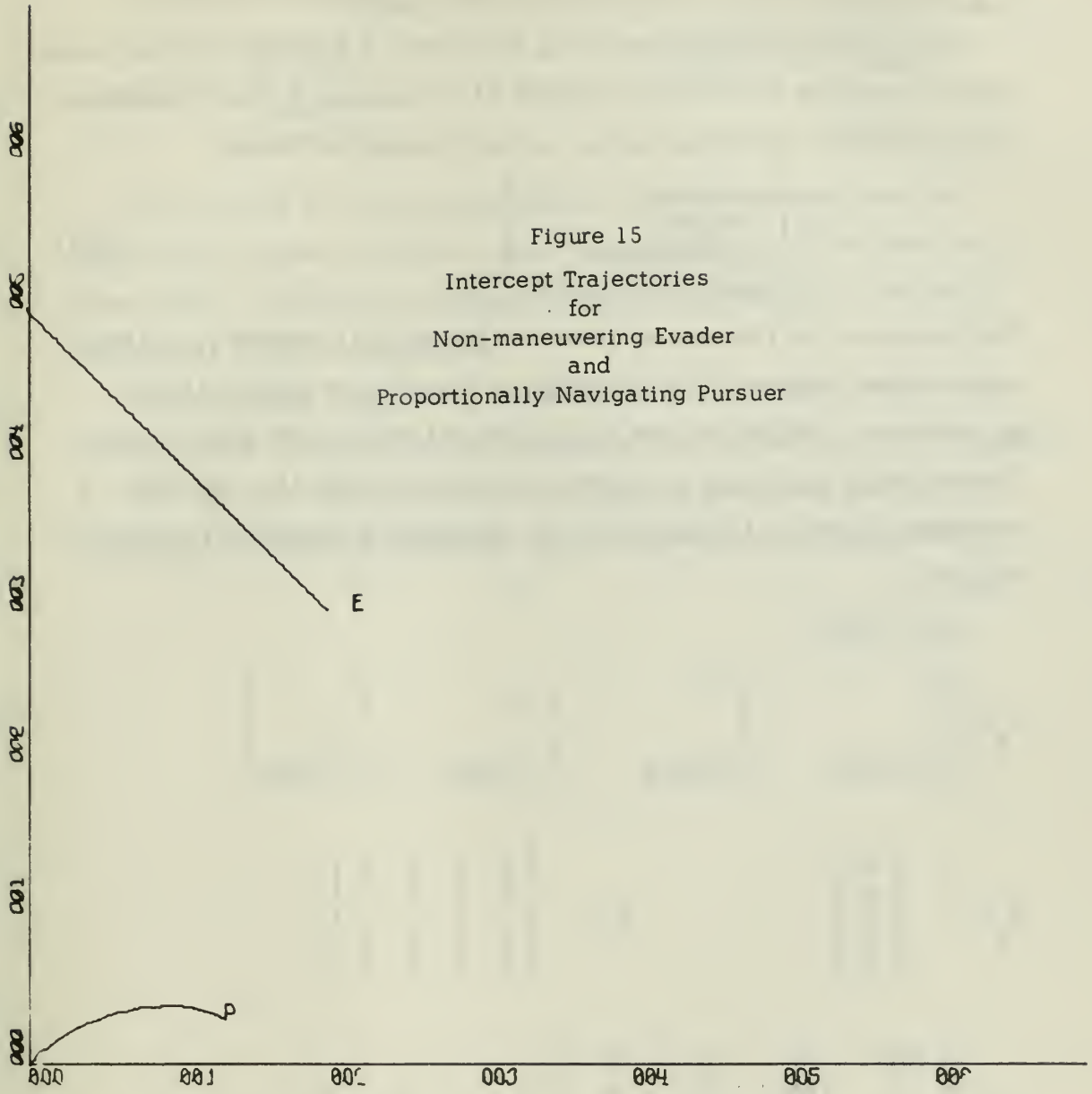
Cost Q = 7.73×10^4

J = 7.73×10^4

The resulting trajectories show vividly the common pitfall of trying to compare two strategies based on different assumptions. Here the fault is in attempting to compare the derived strategy, which allows acceleration in any direction and requires intercept at a specified time, with the strategy of proportional navigation, which assumes the pursuer and evader are restricted to constant speeds (the pursuer faster than the evader) and allows intercept at any time.

Figure 15

Intercept Trajectories
for
Non-maneuvering Evader
and
Proportionally Navigating Pursuer



X-SCALE = 1.00E+01 UNITS/ INCH.
Y-SCALE = 1.00E+02 UNITS/ INCH.

NORTH VS EAST, PURSUIT-EVASION TRAJECTORIES
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 13 2 1 6.7

5.3 Application to the One-sided Control Problem

Run 21 simulates the resulting trajectory of a single system, using evader dynamics to specify the state of the pursuit system desired at the final time. The state vector for the system is defined:

$$\underline{x} = \begin{bmatrix} \text{position} \\ \text{velocity} \\ \text{acceleration} \\ \text{acceleration rate} \end{bmatrix} \tag{52}$$

The dynamics for the pursuit plant are those of an arbitrary, stable, fourth-order system. The dynamics of the evasion plant maintain $\underline{x}_e$ constant. The input data and numerical results are given below. The resulting trajectory is shown in Figures 16 and 17. The time variation of the (1,1) element of the feedback-gain matrix is shown in Figure 18.

Input Data

$$A_p = \begin{bmatrix} 0 & 1.0 & 0 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 1.0 \\ -0.15585 & -0.69454 & -1.56330 & -1.75930 \end{bmatrix}$$

$$B_p = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad A_e = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$Q = \begin{bmatrix} 1000 & 0 & 0 & 0 \\ 0 & 1000 & 0 & 0 \\ 0 & 0 & 1000 & 0 \\ 0 & 0 & 0 & 1000 \end{bmatrix}, \qquad R_p = 1 \quad ,$$

$$\underline{x}_p(0) = \begin{bmatrix} 100 \\ 10 \\ 0 \\ 0 \end{bmatrix} \qquad \underline{x}_e = \begin{bmatrix} 40 \\ 20 \\ 10 \\ 0 \end{bmatrix}$$

$$N = 50 \qquad T = 0.1$$

Numerical Results

$$\underline{z}(N) = \begin{bmatrix} 1.1185 \\ -3.4193 \\ 2.3391 \\ -1.4372 \end{bmatrix}$$

Cost R = 12816
Cost Q = 2048

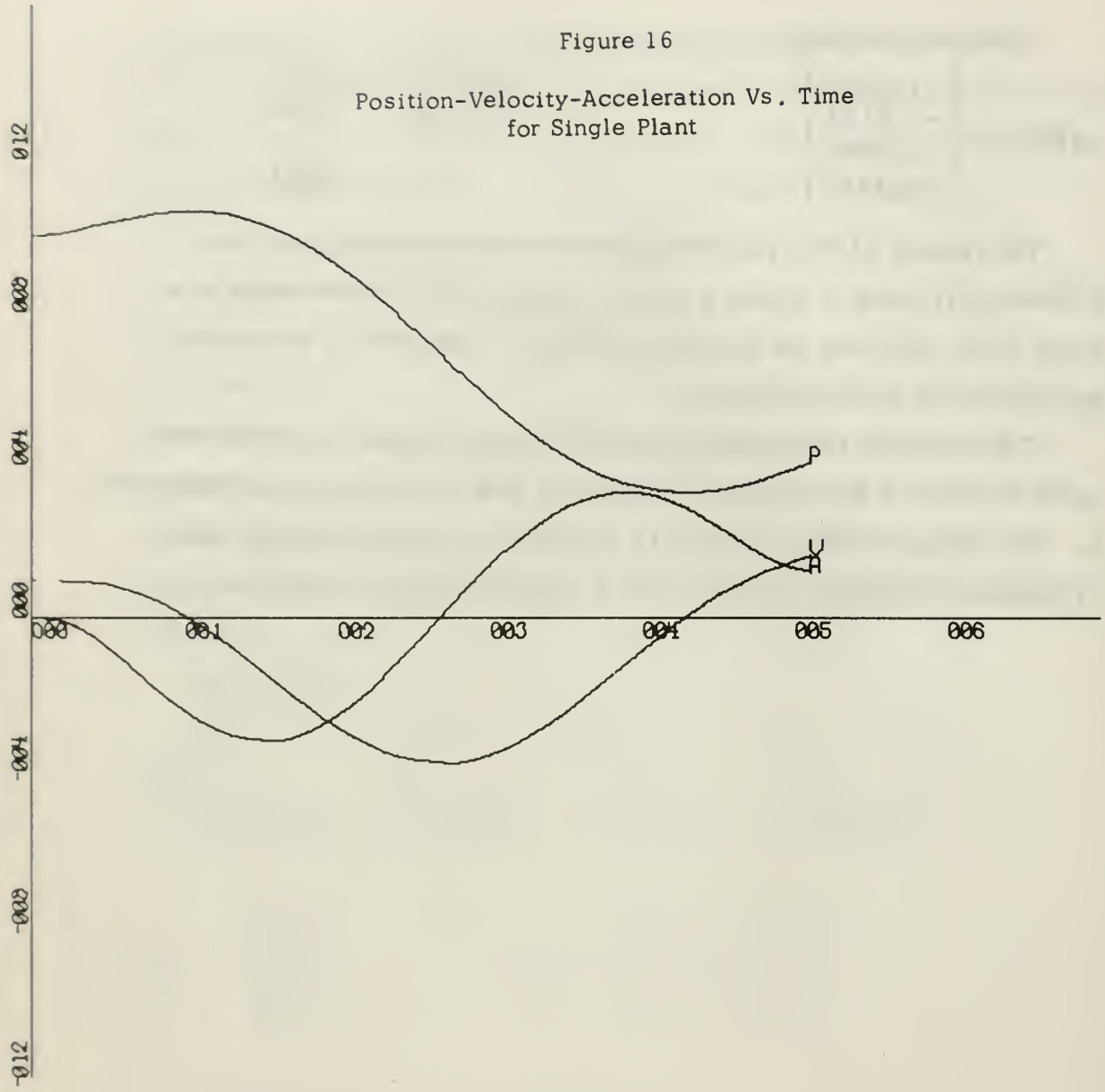
J = 14864

The results of this run demonstrate that the strategy may be successfully used to cause a plant to approach a desired state at a given time, and that the computer program is capable of simulating the results of this application.

The complete input data, showing required format, and the computer printout of the numerical results of this run are given in Appendix B. The changes made in the main program to provide a useful form of graphical computer output for this run are shown in Appendix C.

Figure 16

Position-Velocity-Acceleration Vs. Time
for Single Plant



X-SCALE = 1.00E+00 UNITS/INCH.
Y-SCALE = 4.00E+01 UNITS/INCH.

XP(1,1),XP(2,1),XP(3,1) VS TIME,ONE-SIDED CASE
HUTCHISON,PERMENTER,628,OPTIML.2 RUN 21 2 10 67

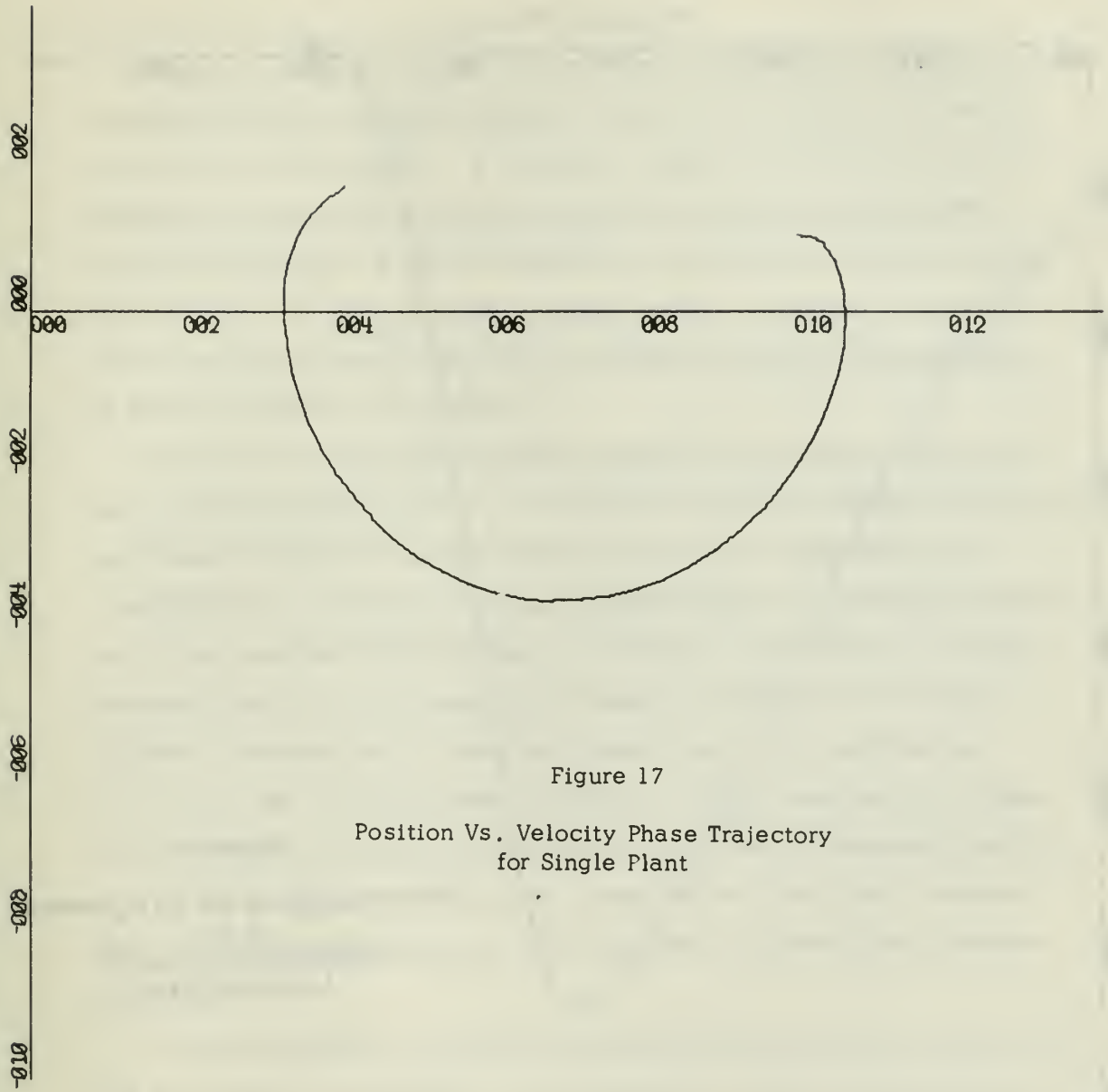


Figure 17

Position Vs. Velocity Phase Trajectory
for Single Plant

X-SCALE = 2.00E+01 UNITS/INCH.
Y-SCALE = 2.00E+01 UNITS/INCH.
VELOCITY VS POSITION, PHASE PLANE, ONE-SIDED CASE
HUTCHISON, PERMENTER, 628, OPTML.2 RUN 21 2 10 67

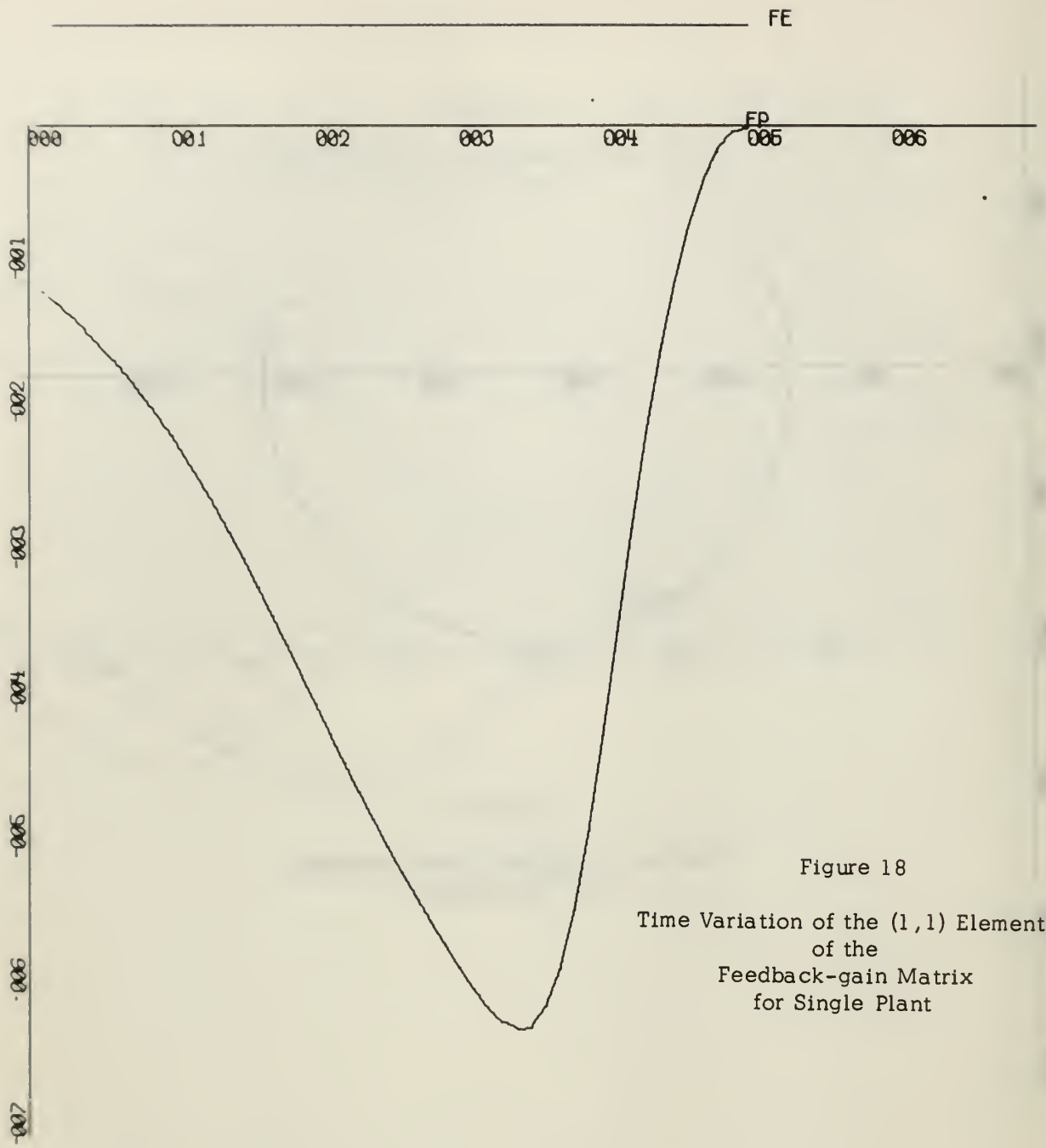


Figure 18

Time Variation of the (1,1) Elements
of the
Feedback-gain Matrix
for Single Plant

X-SCALE = 1.00E+00 UNITS/INCH.
Y-SCALE = 1.00E+00 UNITS/INCH.
FP(1,1) AND FE(1,1) VS TIME
HUTCHISON,PERMENTER,628,OPTML.2 RUN 21 2 10 67

6. CONCLUSIONS

A class of pursuit-evasion games has been solved, illustrating the application of differential game theory and dynamic programming to control system design. A computer program has been provided to simulate the response of linear systems to controls based on the resulting strategy. A set of necessary and sufficient conditions for the existence of the solution has been found. Attempts to simplify these conditions and to provide an adequate physical interpretation of them have been unsuccessful.

That this strategy successfully generates feedback controls for pursuit and evasion plants, for which the system dynamics are linear, and linearly related, and for which some form of "intercept" or "rendezvous" is desired, has been demonstrated. A certain generality of application of the strategy is inherent in the ability to choose different plants for the pursuer and evader, the ability to choose different combinations of weighting matrices in the performance index, and the ability to handle arbitrary initial conditions. If the plant dynamics and performance index are stated beforehand, the F_p, F_e, $\emptyset_p^i$, and $\emptyset_e^i$ matrices may be computed and tabulated, making the "on-line" implementation of the strategy mathematically simple and rapid.

The requirement for a linear relationship between the dynamics of the two plants is restrictive. For example, a linear representation of the dynamics of one aircraft about its roll, pitch, and yaw axes is quite realistic [7]. However, the dynamics of two maneuvering aircraft expressed about their respective axes are not linearly related. A limitation is also imposed on the desired system response by the form of the performance index. For example, an aircraft commander who finds himself on a collision course with an intercepting missile ($\underline{z}(k)^T Q \underline{z}(k) = 0$), may find the application of this strategy, which tells him to conserve his fuel ($\underline{v}(k) = F_e \underline{z}(k) = \underline{0}$), very unsatisfying.

An obvious area for future investigation is the extension of this strategy to include the broader class of pursuit-evasion problems having a performance index of the quadratic form:

$$J = [\underline{x}_p(t_f) - \underline{x}_e(t_f)]^T Q[\underline{x}_p(t_f) - \underline{x}_e(t_f)]$$

$$+ \int_0^{t_f} \{ [\underline{x}_p(t) - \underline{x}_e(t)]^T S[\underline{x}_p(t) - \underline{x}_e(t)]$$

$$+ \underline{u}(t)^T R_p \underline{u}(t) - \underline{v}(t)^T R_e \underline{v}(t) \} dt \qquad (53)$$

Simplification of the necessary and sufficient conditions for the existence of a saddle point and the physical interpretation of these conditions is another area for future investigation. Ho intimates that these conditions are related to the "relative controllability" of the two plants [3], which suggest a method of attack and the possibility of gaining additional insight into the design of pursuit-evasion systems in general.

BIBLIOGRAPHY

1. Ho, Y.C., "Differential Games and Optimal Control Theory", Proceedings of the National Electronics Conference, Vol. 21, pp. 613-615, 1965.

2. Ho, Y.C. and Baron, S., "A Minimal Time Intercept Problem", IEEE Trans. on Automatic Control, Vol. AC-10, No. 2, p. 200, April 1965.

3. Ho, Y.C., Bryson, A.E., and Baron, S., "Differential Games and Optimal Pursuit-Evasion Strategies", IEEE Trans. on Automatic Control, Vol. AC-10, No. 4, pp. 385-389, October 1965.

4. Kuo, B.C., Analysis and Synthesis of Sampled-Data Control Systems, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, pp. 130-143, 1963.

5. Bellman, R.E. and Dreyfus, S.E., Applied Dynamic Programming, Princeton University Press, Princeton, New Jersey, 1962.

6. Ayres, F. Jr., Theory and Problems of Matrices, Schaum Publishing Co., New York, p. 58, 1962.

7. Ellert, F.J. and Merriam, C.W., "Synthesis of Feedback Controls using Optimization Theory - An Example", IEEE Trans. on Automatic Control, Vol. AC-8, No. 2, pp. 89-103, April 1963.

APPENDIX A

DIGITAL COMPUTER PROGRAM FOR TESTING THE OPTIMAL PURSUIT-EVASION STRATEGY

```
-COMP.626,PERGENTER.I/22/0/22/0/0/0/15/23/L/45=54,20,26000,0,OPTIMAL2.
-FIN.L.G.
      PROGRAM OPTIMAL2
C     THIS IS A GENERAL PROGRAM USED TO CONDUCT COMPUTER
C     SIMULATION TESTS OF THE STRATEGY DERIVED IN THE
C     THESIS-
C
C           AN OPTIMAL PURSUIT-EVASION STRATEGY
C           FOR LINEAR SAMPLED-DATA SYSTEMS
C           BY CDR. HUTCHISON AND LT. PERMENTER
C     NOTATION -
C     T     SAMPLED-DATA PERIOD
C     ITAPE    NO. OF SAMPLE PERIODS FOR WHICH STRATEGY
C              IS TO BE RECORDED ON MAGNETIC TAPE
C     IFPFE    NO. OF SAMPLE PERIODS FOR WHICH ELEMENTS
C              OF FP AND FE MATRICES ARE TO BE PLOTTED
C     IPURSUE  NO. OF SAMPLE PERIODS FOR WHICH SIMULATED
C              TRAJECTORIES ARE TO BE RUN
C     NP       ORDER OF THE PURSUIT PLANT
C     MP       DIMENSION OF PURSUIT PLANT CONTROL VECTOR
C     XP       PURSUIT PLANT STATE VECTOR
C     UP       PURSUIT PLANT CONTROL VECTOR
C     AP AND BP (NP*NP) AND (NP*MP) MATRICES DEFINED BY
C              THE CONSTANT COEFFICIENT MATRIX
C              DIFFERENTIAL EQUATION-
C                    XPDOT = AP*XP + BP*UP
C     PHIP     (NP*NP) CONSTANT COEFFICIENT STATE
C              TRANSITION MATRIX FOR PURSUIT PLANT
C     DELP     (NP*MP) CONSTANT COEFFICIENT DISTRIBUTION
C              MATRIX FOR PURSUIT PLANT
C     PHIPI    PHIP**I
C     FP       (MP*NP) PURSUIT PLANT VARIABLE COEFFICIENT
C              FEEDBACK GAIN MATRIX
C     RP       (MP*MP) PURSUIT CONTROL WEIGHTING MATRIX
C     SIMILAR NOTATION IS USED FOR THE EVASION PLANT
C     Z        FINAL STATEMENT DIFFERENCE VECTOR IF NO
```

```
C                    CONTROLS APPLIED
C          Q         (N*N) FINAL STATE WEIGHTING MATRIX
      DIMENSION AP(10,10),AE(10,10),BP(10,10),BE(10,10),
     1PHIP(10,10),PHIE(10,10),DELP(10,10),DELE(10,10),
     2PHIPI(10,10),PHIEI(10,10),XP(10,10),XE(10,10),Q(10,10),
     3RP(10,10),RE(10,10),A1(10,10),A2(10,10),FP(10,10),
     4FE(10,10),Z(10,10),UP(10,10),UE(10,10),IT(12),
     5XPNORTH(900),XENORTH(900),XPEAST(900),XEEAST(900),
     6FPD(900),FED(900),TAU(900)
      REWIND 22
      PRINT 500
  500 FORMAT(1H1,20X,10HINPUT DATA,/)
      READ 501,IT(11)
      READ 501,IT(12)
  501 FORMAT(A8)
      PRINT 502,IT(11),IT(12)
  502 FORMAT(/,20X,18HGRAPH IDENTIFIER  ,A8)
      READ 515,T
  515 FORMAT(E10.4)
      PRINT 516,T
  516 FORMAT(/,20X,E10.4,2X,26HIS THE SAMPLED-DATA PERIOD)
      READ 510,ITAPE
      READ 510,IFPFE
      READ 510,IPURSUE
  510 FORMAT(I10)
      IF(ITAPE-IPURSUE)518,517,517
  517 IF(ITAPE-IFPFE)518,519,519
  518 PRINT 514
  514 FORMAT(/,20X,47HITAPE MUST BE ATLEAST EQUAL TO IFPFE OR IPURSUE)
      GO TO 999
  519 PRINT 511,ITAPE
  511 FORMAT(/,20X,I4,2X,33HSAMPLES WILL BE GENERATED ON TAPE)
      PRINT 512,IFPFE
  512 FORMAT(/,20X,I4,2X,39HSAMPLES WILL BE USED FOR FP AND FE PLOT)
      PRINT 513,IPURSUE
```

```
  515 FORMAT(/,20X,I4,2X,14HSAMPLES WILL BE USED FOR TRAJECTORIES PLOT)
      CALL MAURICE(AP,NP,NP)
      NA=8HAP
      CALL JOHANN(AP,NP,NP,NA)
      CALL MAURICE(BP,NP,MP)
      NA=8HBP
      CALL JOHANN(BP,NP,MP,NA)
      CALL MAURICE(AE,NE,NE)
      NA=8HAE
      CALL JOHANN(AE,NE,NE,NA)
      CALL MAURICE(BE,NE,ME)
      NA=8HBE
      CALL JOHANN(BE,NE,ME,NA)
      CALL MAURICE(Q,NP,NP)
      NA = 8HQ
      CALL JOHANN(Q,NP,NP,NA)
      CALL MAURICE(RP,MP,MP)
      NA = 8HRP
      CALL JOHANN(RP,MP,MP,NA)
      CALL MAURICE(RE,ME,ME)
      NA = 8HRE
      CALL JOHANN(RE,ME,ME,NA)
      CALL MAURICE(XP,NP,1)
      NA=8HXP(ZERO)
      CALL JOHANN(XP,NP,1,NA)
      CALL MAURICE(XE,NE,1)
      NA = 8HXE(ZERO)
      CALL JOHANN(XE,NE,1,NA)
C          INPUT AND RECORDING OF DATA CARDS AND PLANT PARAMETERS
C          IS NOW COMPLETE
      PRINT 501
  501 FORMAT(1H1,20X,18HOUTPUT INFORMATION,/)
      CALL PHIDEL(AP,BP,NP,MP,T,PHIP,DELP)
      NA=8HPHIP
      CALL JOHANN(PHIP,NP,NP,NA)
```

```
      NA=8HDELP
      CALL JOHANN(DELP,MP,MP,NA)
      CALL PHIDEL(AE,BE,NE,ME,T,PHIE,DELE)
      NA=8HPHIE
      CALL JOHANN(PHIE,NE,NE,NA)
      NA=8HDELE
      CALL JOHANN(DELE,NE,ME,NA)
      N = NE
  520 CALL OPTCON2(PHIP,PHIE,DELP,DELE,Q,RP,RE,ITAPE,N,MP,ME)
  551 J=IFPFE-1
      DO 503 I=1,J
      CALL LOCATE2(FP,MP,N,K)
      CALL LOCATE2(FE,ME,N,K)
      CALL LOCATE2(PHIPI,N,N,K)
  503 CALL LOCATE2(PHIEI,N,N,K)
      DO 100 I=1,IFPFE
      CALL LOCATE2(FP,MP,N,K)
      CALL LOCATE2(FE,ME,N,K)
      CALL LOCATE2(PHIPI,N,N,K)
      CALL LOCATE2(PHIEI,N,N,K)
      IBACK=IFPFE-I+1
      PRINT 571,IBACK
  571 FORMAT(/,20X,15HSAMPLE NUMBER  ,I3,/)
      NA = 8HFP
      CALL JOHANN(FP,MP,N,NA)
      NA = 8HFE
      CALL JOHANN(FE,ME,N,NA)
      NA = 8HPHIPI
      CALL JOHANN(PHIPI,N,N,NA)
      NA = 8HPHIEI
      CALL JOHANN(PHIEI,N,N,NA)
      IF(K-1) 506,506,504
  504 DO 505 J=1,8
  505 BACKSPACE 22
  506 T1=I
```

```
      TAU(I)=I*T1
      FED(I)=FE(1,1)
      FPD(I)=FP(1,1)
  100 CONTINUE
      LAST = 0
      IT(1)=8HFP(1,1)
      IT(2)=8HAND FE(1
      IT(3)=8H,1) VS T
      IT(4)=8HIME
      IT(5)=8H
      IT(6)=8H
      IT(7) =8HHUTCHISO
      IT(8) =8HN,PERMEN
      IT(9) =8HTER,620,
      IT(10)=8HOPTML.2
      LABEL=4HFP
      CALL DRAW(IFPFE,TAU,FPD,1,0,LABEL,IT,0,0,0,0,1,1,7,7,0,LAST)
      LABEL=4H  FE
      CALL DRAW(IFPFE,TAU,FED,3,0,LABEL,IT,0,0,0,0,1,1,7,7,0,LAST)
      REWIND 22
      COSTR = 0.0
  554 J=IPURSUE-1
      DO 601 I=1,J
      CALL LOCATE2(FP,MP,NP,K)
      CALL LOCATE2(FE,ME,NE,K)
      CALL LOCATE2(PHIPI,NP,NP,K)
      CALL LOCATE2(PHIEI,NE,NE,K)
  601 CONTINUE
C         *** VARIOUS STATEMENT CHANGES MAY BE REQUIRED
C         *** BELOW FOR DIFFERENT PLANTS
      XPNORTH(1)=XP(1,1)
      XPEAST(1)=XP(3,1)
      XENORTH(1)=XE(1,1)
      XEEAST(1)=XE(3,1)
      JPURSUE=IPURSUE+1
```

```
      DO 602 INDEX=2,JPOINT
      CALL LOCATE2(FP,MP,NP,K)
      CALL LOCATE2(FE,ME,NE,K)
      CALL LOCATE2(PHIPI,NP,NP,K)
      CALL LOCATE2(PHIEI,NE,NE,K)
      IF(K-1)603,603,604
  604 DO 605 I=1,8
  605 BACKSPACE 22
  603 CALL PROD(PHIPI,XP,Z,NP,NP,1)
      CALL PROD(PHIEI,XE,A1,NE,NE,1)
      CALL SUBTRAC(Z,A1,Z,NE,1)
      CALL PROD(FP,Z,UP,MP,NP,1)
      CALL PROD(FE,Z,UE,ME,NE,1)
C         *** INSERT STATEMENTS HERE TO PROVIDE DIFFERENT
C         *** CONTROL STRATEGIES FOR PURSUIT OR EVADER
      CALL PLANT(PHIP,DELP,XP,UP,NP,MP)
      CALL PLANT(PHIE,DELE,XE,UE,NE,ME)
      XPNORTH(INDEX)=XP(1,1)
      XPEAST(INDEX)=XP(3,1)
      XENORTH(INDEX)=XE(1,1)
      XEEAST(INDEX)=XE(3,1)
      CALL TRANS(UP,A1,MP,1)
      CALL PROD(A1,RP,A2,1,MP,MP)
      CALL PROD(A2,UP,A1,1,MP,1)
      COSTR=COSTR+A1(1,1)
      CALL TRANS(UE,A1,ME,1)
      CALL PROD(A1,RE,A2,1,ME,ME)
      CALL PROD(A2,UE,A1,1,ME,1)
      COSTR=COSTR-A1(1,1)
  602 CONTINUE
      PRINT 200
  200 FORMAT(1H1,10X,25HTERMINAL PERFORMANCE DATA,/)
      CALL SUBTRAC(XP,XE,Z,NP,1)
      NA = 8HXP-XE
      CALL JOHANN(Z,NP,1,NA)
```

```
      CALL TRANS(Z,A1,NP,1)
      CALL PROD(A1,G,A2,1,NP,NP)
      CALL PROD(A2,Z,A1,1,NP,1)
      PRINT 201, A1(1,1)
  201 FORMAT(/,20X,41HPERFORMANCE INDEX DUE TO STATES          ,E11.4,/)
      PRINT 202,COSTR
  202 FORMAT(/,20X,41HPERFORMANCE INDEX DUE TO CONTROL EFFORT  ,E11.4,/)
      TOTAL=A1(1,1)+COSTR
      PRINT 203,TOTAL
  203 FORMAT(/,20X,41HTOTAL PERFORMANCE INDEX                  ,E11.4,/)
      LAST = 0
      IT(1)=8HNORTH VS
      IT(2)=8H EAST,PS
      IT(3)=8HRSUIT-EV
      IT(4)=8HASION TR
      IT(5)=8HAJECTORI
      IT(6)=8HES
      SCALE = 100.0
      LABEL=4H  E
      CALL DRAW(JPURSUE,XEEAST,XENORTH,1,0,LABEL,IT,SCALE,SCALE,0,0,
     10,6,7,7,0,LAST)
      LABEL=4HP
      CALL DRAW(JPURSUE,XPEAST,XPNORTH,3,0,LABEL,IT,SCALE,SCALE,0,0,
     10,6,7,7,0,LAST)
  999 REWIND 22
      END
```

```
      SUBROUTINE JOHANN(OUT,L,M,INAME)
C         JOHANN    JOHANN GUTENBERG, 15TH CENTURY GERMAN PRINTER
C       PRINTS THE OUT MATRIX, AND ITS NAME
      DIMENSION OUT(10,10)
      PRINT 1,L,M,INAME
      DO 2 IR=1,L
    2 PRINT 3,(OUT(IR,IC),IC=1,M)
    1 FORMAT(/,10X,I2,2H *,I2,3X,A8)
    3 FORMAT(6X,10E11.4)
      END


      SUBROUTINE MAURICE(ENTER,L,M)
C         MAURICE= MAURICE EVANS, ACTOR, READER OF MATRICES FROM DATA CARDS
      DIMENSION ENTER(10,10)
      READ 1,L,M
    1 FORMAT(2I2)
      DO 2 IR=1,L
    2 READ 3,(ENTER(IR,IC),IC=1,M)
    3 FORMAT(10F10.3)
      END


      SUBROUTINE LOCATE2(OUT,L,M,N)
C         LOCATES MATRIX STORED ON TAPE BY  SUBROUTINE RECORD2
      DIMENSION OUT(10,10)
      READ TAPE 22,N,((OUT(I,J),J=1,M),I=1,L)
      END


      SUBROUTINE RECORD2(ENTER,L,M,N)
C         STORES A MATRIX AND ITS SEQUENCE NUMBER ON TAPE IN BINARY FORM
      DIMENSION ENTER(10,10)
      WRITE TAPE 22,N,((ENTER(I,J),J=1,M),I=1,L)
      END
```

```
      SUBROUTINE OPTCON(PHIP,PHIE,DELTAP,DELTAE,
     1Q,RP,RE,ISAMPLE,N,MP,ME)
C          REFERENCE - AN OPTIMAL PURSUIT-EVASION STRATEGY
C                      FOR LINEAR SAMPLED-DATA SYSTEMS
C          THIS SUBROUTINE PRODUCES THE FE AND FP MATRICES IN INVERSE ORDER
C          FROM THE ORDER THAT THEY ARE UTILIZED IN THE MAIN PROGRAM
C          SYMBOLOGY
C          ISAMPLE    NUMBER OF FE OR FP MATRICES TO BE GENERATED
C          E          EVADER POSTSCRIPT
C          P          PURSUER POSTSCRIPT
C          DELTAP     PURSUIT PLANT STATE DISTRIBUTION MATRIX
C                     FOR ONE SAMPLE PERIOD
C          DELP       PHIP!*DELTAP
C          PHIP       PURSUIT PLANT STATE TRANSITION MATRIX
C                     FOR ONE SAMPLE PERIOD
C          PHIPI      PURSUIT PLANT STATE TRANSITION MATRIX
C                     FOR I SAMPLE PERIODS
C          RE AND RP MAY BE INTRODUCED EACH SAMPLE INSTEAD OF USING CONSTANT
C          MATRICES.  THEIR POSITIONS FOR PROGRAM ENTRY ARE LATER NOTED.
      DIMENSION PHIP(10,10),PHIE(10,10),DELTAP(10,10),DELTAE(10,10),
     1Q(10,10),RP(10,10),RE(10,10),A1(10,10),A2(10,10),A3(10,10),
     2A4(10,10),P(10,10),PSI(10,10),DELP(10,10),DELE(10,10),FP(10,10),
     3FE(10,10),PHIPI(10,10),PHIEI(10,10)
      REWIND 22
      CALL EQUAL(Q,P,N,N)
      CALL EQUAL(DELTAP,DELP,N,MP)
      CALL EQUAL(DELTAE,DELE,N,ME)
      UNITY=1.0
      CALL IDENTIT(PHIPI,N,UNITY)
      CALL IDENTIT(PHIEI,N,UNITY)
C          INITIALIZATION IS NOW COMPLETE
      DO 1 INUM=1,ISAMPLE
C          NEXT FORM THE PARTITIONED F MATRIX       F = C * B
C          SEE REFERENCE EQUATION (43D)
      CALL TRANS(DELP,A1,N,MP)
```

```
      CALL TRANS(DELE,A2,N,ME)
      CALL PROD(A1,P,A3,MP,N,N)
      CALL MINUS(A3,A4,MP,N)
      CALL PROD(A2,P,A1,ME,N,N)
      DO 2 I=1,ME
      DO 2 J=1,N
      K=I+MP
    2 A4(K,J)=A1(I,J)
      MPME = MP+ME
C          A4 = THE D MATRIX, PARTITIONED INTO TOP AND BOTTOM PARTS
C          SEE REFERENCE EQUATION (43D)
      CALL PROD(A3,DELP,A1,MP,N,MP)
C          *** INSERT STATEMENTS HERE TO READ OR GENERATE
C          *** NEW RP(N-I) AND RE(N-I) IF DESIRED
      CALL SUM(A1,RP,A1,MP,MP)
      CALL DEFINITE(A1,MP,IFLAG)
      IF(IFLAG) 100,102,100
  100 PRINT 101, INUM
      NA=8HA1
      CALL JOHANN(A1,MP,MP,NA)
  101 FORMAT(/,10X,9H***INUM =,I5,5X,30HTEST FOR OPTIMAL PURSUIT FAILS)
  102 CALL PROD(A4,DELE,A2,MP,N,ME)
      DO 4 I=1,MP
      DO 4 J=1,ME
      L=J+MP
    4 A1(I,L)=A2(I,J)
      CALL TRANS(DELE,A2,N,ME)
      CALL PROD(A2,P,A3,ME,N,N)
      CALL PROD(A3,DELP,A2,ME,N,MP)
      CALL MINUS(A2,A2,ME,MP)
      DO 3 I=1,ME
      DO 3 J=1,MP
      K=I+MP
    3 A1(K,J)=A2(I,J)
      CALL PROD(A3,DELE,A2,ME,N,ME)
```

```
      CALL SUBTRAC(A2,RE,A2,ME,ME)
      DO 5 I=1,ME
      DO 5 J=1,ME
      K=I+MP
      L=J+MP
    5 A1(K,L)=A2(I,J)
      CALL MINUS(A2,A2,ME,ME)
      CALL DEFINITE(A2,ME,IFLAG)
      IF (IFLAG) 103,105,103
  103 PRINT 104, INUM
      CALL MINUS(A2,A2,ME,ME)
      NA=8HA2
      CALL JOHANN(A2,ME,ME,NA)
  104 FORMAT(/,10X,9H***INUM =,I5,5X,30HTEST FOR OPTIMAL EVASION FAILS)
  105 CALL INVERSE(A1,A3,MPME,SMALL)
      IF(SMALL)20,21,21
   20 PRINT 22, INUM
   22 FORMAT(/,10X,7HINUM = ,I5)
   21 CONTINUE
C          A3 = THE C MATRIX MENTIONED ABOVE - SEE EQUATION (43D)
      CALL PROD(A3,A4,A1,MPME,MPME,N)
C          A1 IS THE LEFT-HAND SIDE OF EQUATION (43D)
C          THE F MATRIX MUST NEXT BE SEPARATED INTO FP AND FE.
      CALL EQUAL(A1,FP,MP,N)
C          FP MATRIX IS NOW SEPARATED AS UPPER PORTION OF THE F MATRIX.
      DO 6 I=1,ME
      DO 6 J=1,N
      K=I+MP
    6 FE(I,J)=A1(K,J)
C          FE MATRIX IS THE LOWER PORTION OF F.
      CALL PROD(PHIPI,PHIP,A1,N,N,N)
      CALL EQUAL(A1,PHIPI,N,N)
      CALL PROD(PHIEI,PHIE,A1,N,N,N)
      CALL EQUAL(A1,PHIEI,N,N)
      CALL RECORD2(FP,MP,N,INUM)
```

```
      CALL RECORD2(FE,ME,N,INUM)
      CALL RECORD2(PHIPI,N,N,INUM)
      CALL RECORD2(PHIEI,N,N,INUM)
C         FP(INUM), FE(INUM), PHIPI(INUM), AND PHIEI(INUM)
C         ARE NOW STORED ON TAPE 22, BINARY FORM.
      CALL PROD(DELP,FP,A1,N,MP,N)
      CALL PROD(DELE,FE,A2,N,ME,N)
      CALL SUBTRAC(A1,A2,A3,N,N)
      CALL IDENTIT(A2,N,UNITY)
      CALL SUM(A3,A2,PSI,N,N)
C         PSI(I) NOW COMPLETE SEE EQUATION (43E)
      CALL TRANS(PSI,A1,N,N)
      CALL PROD(A1,P,A2,N,N,N)
      CALL PROD(A2,PSI,A1,N,N,N)
      CALL TRANS(FP,A2,MP,N)
      CALL PROD(A2,RP,A3,N,MP,MP)
      CALL PROD(A3,FP,A2,N,MP,N)
      CALL TRANS(FE,A3,ME,N)
      CALL PROD(A3,RE,A4,N,ME,ME)
      CALL PROD(A4,FE,A3,N,ME,N)
      CALL SUM(A1,A2,A4,N,N)
      CALL SUBTRAC(A4,A3,P,N,N)
C         P(I) NOW COMPLETE SEE EQUATION (43F)
      CALL PROD(PHIP,DELP,A1,N,N,MP)
      CALL EQUAL(A1,DELP,N,MP)
C         DELP(I) NOW COMPLETE SEE EQUATION (43G)
      CALL PROD(PHIE,DELE,A2,N,N,ME)
    1 CALL EQUAL(A2,DELE,N,ME)
C         DELE(I) NOW COMPLETE SEE EQUATION (43H)
      END FILE 22
      REWIND 22
C         SINCE END OF FILE IS A RECORD, 4*ISAMPLE+1 RECORDS ARE ON TAPE
      PRINT 110
  110 FORMAT(/,20X,16HOPTCON2 COMPLETE)
      END
```

```
      SUBROUTINE PHIDEL(A,B,N,M,T,PHI,DEL)
C     PHI AND DELTA COMPUTATION
C         GIVEN THE LINEAR TIME INVARIANT DIFFERENTIAL EQUATION
C           XDOT = A * X  +  B * U
C         THIS PROGRAM FINDS PHI AND DELTA IN THE DIFFERENCE EQUATION
C           X(K+1)  =   PHI * X(K)  +  DEL * U(K)
C         SYMBOLOGY
C         XDOT        N*1 COLUMN VECTOR
C         A           N*N CONSTANT COEFFICIENT MATRIX
C         X           N*1 COLUMN VECTOR OF SYSTEM STATES
C         B           N*M CONSTANT COEFFICIENT MATRIX
C         U           M*1 COLUMN MATRIX OF INPUTS
C         K           POSITION NUMBER OF A GIVEN SAMPLE
C         PHI         N*N CONSTANT COEFFICIENT STATE TRANSITION MATRIX
C         DEL         N*M CONSTANT COEFFICIENT DISTRIBUTION MATRIX
C         DEN         FACTORIAL DENOMINATOR IN A SUMMATION TERM
C         T           DATA SAMPLE PERIOD
      DIMENSION A(10,10),B(10,10),PHI(10,10),DEL(10,10),A1(10,10),
     1A2(10,10),A3(10,10)
      UNITY=1.0
      CALL IDENTIT(PHI,N,UNITY)
      CALL IDENTIT(A1,N,UNITY)
      CALL IDENTIT(DEL,N,T)
      CALL PROD(DEL,A,A2,N,N,N)
      CALL EQUAL(A2,A,N,N)
      DEN=1.0
      Z=0.0
   10 Z=Z+1.
      DEN=DEN*Z
      CALL PROD(A,A1,A2,N,N,N)
      CALL EQUAL(A2,A1,N,N)
      DENPHI=1./DEN
      CALL IDENTIT(A2,N,DENPHI)
      CALL PROD(A2,A1,A3,N,N,N)
      CALL MAXIMUM (A3,N,N,TERM)
```

```
      CALL SUM(A3,PHI,A2,N,N)
      CALL EQUAL(A2,PHI,N,N)
C         PHI IS NOW COMPLETE FOR INUM ITERATIONS
      DENDEL=DENPHI*T/(Z+1.)
      CALL IDENTIT(A2,N,DENDEL)
      CALL PROD(A2,A1,A3,N,N,N)
      CALL SUM(A3,DEL,A2,N,N)
      CALL EQUAL(A2,DEL,N,N)
C         DEL IS NOW COMPLETE, EXCEPT FOR MULTIPLICATION BY B,
C         FOR INUM ITERATIONS
      SIZE=.1E-20
      IF(TERM-SIZE)20,20,10
   20 CALL PROD(DEL,B,A2,N,N,M)
      CALL EQUAL(A2,DEL,N,M)
      NUM=Z
      PRINT 30, NUM,SIZE,T
   30 FORMAT(/,10X,25HSUBROUTINE PHIDEL  NUM = ,I4,4X,7HSIZE = ,E10.4,
     14X,4HT = ,E10.4)
      END
```

```
      SUBROUTINE DEFINITE(C,ENTER,N,IFLAG)
      DIMENSION ENTER(10,10),A(10,10),B(10,10),C(10,10),D(10,10)
C         ENTER IS A POSITIVE DEFINITE SYMMETRIC MATRIX PROVIDED
C         IT CAN BE EXPRESSED BY THE OPERATION
C              ENTER = A * B
C         WHERE     A = A LOWER TRIANGULAR MATRIX
C                   B = A(TRANSPOSED)
C         THE INDICATOR IS RETURNED -
C                    IFLAG = 0     ENTER IS POSITIVE DEFINITE
C                    IFLAG = 1     ENTER NOT POSITIVE DEFINITE
      IFLAG = 0
      DO 1 I=1,N
      DO 1 J=1,N
    1 C(I,J) = 0.0
      DO 10 K =1,N
      CALL SUBTRAC(ENTER,C,D,N,N)
      IF(D(K,K)) 2,2,3
    2 IFLAG = 1
      RETURN
    3 X = SQRTF(D(K,K))
      DO 4 I=1,N
      A(I,K) = D(I,K)/X
    4 B(K,I) = A(I,K)
   10 CALL PROD(A,B,C,N,K,N)
      END


      SUBROUTINE EQUAL(ENTER,OUT,M,N)
C         THE OUT MATRIX IS SET EQUAL TO THE ENTER MATRIX
      DIMENSION ENTER(10,10),OUT(10,10)
      DO 1 IR=1,M
      DO 1 IC=1,N
    1 OUT(IR,IC)=ENTER(IR,IC)
      END
```

```
      SUBROUTINE INVERSE(B,X,N,SMALL)
C          B = INPUT MATRIX      X = OUTPUT MATRIX
C          THE DIMENSIONS OF INVERSE MUST BE THE SAME AS THOSE PROVIDED IN THE
C          CALLING PROGRAM FOR THE MATRIX TO BE INVERTED.  THE FOLLOWING
C          IS NOT A LEGAL CALL FOR INVERSE.  SEE SUBROUTINE PROD.
C          CALL INVERSE(ALPHA,ALPHA,M,SMALL)
      DIMENSION A(10,10),B(10,10),X(10,10)
      SMALL=1.E-8
C          SMALL IS A FUNCTION OF THE COMPUTER USED.
C          IT IS SET HERE FOR THE CDC 1604 AT NAVAL POSTGRADUATE SCHOOL
      EP=SMALL
      DO 1 I=1,N
      DO 1 J=1,N
      A(I,J)=B(I,J)
    1 X(I,J)=0.0
      DO 2 K=1,N
    2 X(K,K)=1.0
   10 DO 34 L=1,N
      KP=
      Z=0.0
      DO 12 K=L,N
      IF(Z-ABSF(A(K,L)))11,12,12
   11 Z=ABSF(A(K,L))
      KP = K
   12 CONTINUE
      IF(L-KP)13,20,20
   13 DO 14 J=L,N
      Z=A(L,J)
      A(L,J)=A(KP,J)
   14 A(KP,J)=Z
      DO 15 J=1,N
      Z=X(L,J)
      X(L,J)=X(KP,J)
   15 X(KP,J)=Z
   20 IF(ABSF(A(L,L))-EP)50,50,30
```

```
      30 IF(L-N)31,34,34
      31 LP1=L+1
         DO 36 K=LP1,N
         IF(A(K,L))32,36,32
      32 RATIO=A(K,L)/A(L,L)
         DO 33 J=LP1,N
      33 A(K,J)=A(K,J)-RATIO*A(L,J)
         DO 35 J=1,N
      35 X(K,J)=X(K,J)-RATIO*X(L,J)
      36 CONTINUE
      34 CONTINUE
      40 DO 43 I=1,N
         II=N+1-I
         DO 43 J=1,N
         S=0.0
         IF(II-N)41,43,43
      41 IIP1=II+1
         DO 42 K=IIP1,N
      42 S=S+A(II,K)*X(K,J)
      43 X(II,J)=(X(II,J)-S)/A(II,II)
         KER=1
         RETURN
      50 KER=2
         PRINT 100, SMALL
     100 FORMAT(/,10X,43HSUBROUTINE INVERSE INPUT SINGULAR, SMALL = ,E10.5)
         SMALL = - 1.0
C            SMALL IS MADE NEGATIVE IN THE EVENT OF A SINGULAR
C            MATRIX.  IT IS THEN THE FLAG FOR THE CALLING PROGRAM
C            TO USE IF NEEDED.
         NA = 8HINPUT
         CALL JOHANN(B,N,N,NA)
         NA = 8HINVERSE
         CALL JOHANN(X,N,N,NA)
         END
```

```
      SUBROUTINE IDENTIT(OUT,L,ALTER)
C         FORMS AN IDENTITY MATRIX TIMES A CONSTANT, ALTER
      DIMENSION OUT(10,10)
      DO 1 IR=1,L
      DO 1 IC=1,L
      OUT(IR,IC)=0.0
    1 OUT(IR,IR)=ALTER
      END


      SUBROUTINE MAXIMUM(ENTER,M,N,AMAX)
C         AMAX = ABSOLUTE VALUE OF MAXIMUM MAGNITUDE TERM IN ENTER MATRIX
      DIMENSION ENTER(10,10)
      AMAX=0.0
      DO 2 IR=1,M
      DO 2 IC=1,N
      IF(AMAX-ABSF(ENTER(IR,IC)))1,2,2
    1 AMAX=ABSF(ENTER(IR,IC))
    2 CONTINUE
      END


      SUBROUTINE MINUS(ENTER,OUT,L,M)
C         OUT MATRIX IS THE ENTER MATRIX WITH SIGNS REVERSED
      DIMENSION ENTER(10,10),OUT(10,10)
      DO 3 I=1,L
      DO 3 J=1,M
    3 OUT(I,J)=-ENTER(I,J)
      END
```

```
      SUBROUTINE PLANT(PHI,DEL,X,U,N,M)
      DIMENSION A1(10,10),PHI(10,10),DEL(10,10),X(10,10),
     1U(10,10)
      CALL PROD(PHI,X,A1,N,N,1)
      CALL PROD(DEL,U,X,N,M,1)
      CALL SUM(X,A1,X,N,1)
      END


      SUBROUTINE PROD(ENTER,ENTRE,OUT,L,M,N)
C         MATRIX PRODUCT OUT(L*N) = ENTER(L*M) * ENTRE(M*N)
C         THE FOLLOWING IS NOT A LEGAL CALL FOR PROD
C         CALL PROD(ALPHA,BETA,ALPHA,N,M,L)  THE OUTPUT MATRIX MAY NOT
C         BE NAMED THE SAME AS ONE OF THE INPUT MATRICES.
      DIMENSION ENTER(10,10),ENTRE(10,10),OUT(10,10)
      DO 5 I=1,L
      DO 5 K=1,N
      OUT(I,K)=0.0
      DO 5 J=1,M
    5 OUT(I,K)=OUT(I,K)+ENTER(I,J)*ENTRE(J,K)
      END


      SUBROUTINE SUBTRAC(ENTER,ENTRE,OUT,L,M)
C         MATRIX SUBTRACTION   OUT = ENTER - ENTRE
      DIMENSION ENTER(10,10),ENTRE(10,10),OUT(10,10)
      DO 3 I=1,L
      DO 3 J=1,M
    3 OUT(I,J)=ENTER(I,J)-ENTRE(I,J)
      END
```

```
      SUBROUTINE SUM(ENTER,ENTRE,OUT,L,M)
C         MATRIX ADDITION  OUT = ENTER + ENTRE
      DIMENSION ENTER(10,10),ENTRE(10,10),OUT(10,10)
      DO 3 I=1,L
      DO 3 J=1,M
    3 OUT(I,J)=ENTER(I,J)+ENTRE(I,J)
      END


      SUBROUTINE TRANS(ENTER,OUT,L,M)
C         MATRIX TRANSPOSITION  ENTER(L*M) BECOMES OUT(M*L)
      DIMENSION ENTER(10,10),OUT(10,10)
      DO 3 I=1,L
      DO 3 J=1,M
    3 OUT(J,I)=ENTER(I,J)
      END
```

APPENDIX B

SAMPLE FORMAT FOR COMPUTER INPUT DATA AND COMPUTER PRINTED OUTPUT

```
1234567890123456789012345678901234567890123456789012345678901234567890
                    INPUT DATA FOR RUN 1 FOLLOWS
RUN 1
  2 10 67
    1.0    T
                20      ITAPE
                10      IFPFE
                20      IPURSUE
          404       AP
        .0          1.0         .0          .0
        .0          .0          .0          .0
        .0          .0          .0          1.0
        .0          .0          .0          .0
          402       BP
        .0          .0
        .0          .0
        1.0         .0
        .0          1.0
          404       AE
        .0          1.0         .0          .0
        .0          .0          .0          .0
        .0          .0          .0          -1.0
        .0          .0          .0          .0
          402       BE
        .0          .0
        .0          .0
        1.0         .0
        .0          1.0
          404       Q
        1.0         .0          .0          .0
        .0          .0          .0          .0
        .0          .0          1.0         .0
        .0          .0          .0          .0
```

```
 202        RP
1.0         .0
.0          1.0
 202        RE
5.0         .0
.0          5.0
 401        XP(0)
.0
5.0
.0
5.0
 401        XE(0)
500.
-10.
.0
10.
```

INPUT DATA

GRAPH IDENTIFIER RUN 1

GRAPH IDENTIFIER 2 10 67

1.CCCCE CC IS THE SAMPLED-DATA PERIOD

2C SAMPLES WILL BE GENERATED ON TAPE

1C SAMPLES WILL BE USED FOR EP AND FE PLOT

2C SAMPLES WILL BE USED FOR TRAJECTORIES PLOT

```
    4 * 4    AP
         0 1.CCCCE 00          0           C
         C          C          0           C
         0          C          C 1.CCCCE 0C
         0          C          C           C

    4 * 2    BP
         0          C
1.CCCCE C0          C
         C          C
         C 1.CCCCE CC

    4 * 4    AE
         0 1.CCCCE 0C          0           C
         C          C          0           C
         0          C          C 1.CCCCE CC
         0          0          C           C

    4 * 2    BE
         0          C
1.CCCCE C0          0
         C          C
         0 1.CCCCE CC

    4 * 4    C
1.CCCCE CC          0          0           0
         0          C          C           C
         C          0 1.CCCCE CC           C
         C          0          0           C
```

```
      2 * 2    RP
1.CCCCE CC
         C 1.CCCCE CC

      2 * 2    RE
5.CCCCF CC
         C 5.CCCCE CC

      4 * 1    XP(ZERO)
         C
5.CCCCF CC
         C
5.CCCCE CC

      4 * 1    XE(ZERO)
5.CCCCF C2
-1.CCCCF C1
         C
1.CCCCE C1
```

```
                    OUTPUT INFORMATION

    SUBROUTINE PHIDEL  NUM =     2     SIZE = 1.0000E-21     T = 1.0000E 00

     4 * 4    PHIP
1.0000E 00 1.0000E 00            0             0
           0 1.0000E 00            0             0
           0             0 1.0000E 00 1.0000E 00
           0             0            0 1.0000E 00

     4 * 2    DELP
5.0000E-01             0
1.0000E 00             0
           0 5.0000E-01
           0 1.0000E 00

    SUBROUTINE PHIDEL  NUM =     2     SIZE = 1.0000E-21     T = 1.0000E 00

     4 * 4    PHIE
1.0000E 00 1.0000E 00            0             0
           0 1.0000E 00            0             0
           0             0 1.0000E 00 1.0000E 00
           0             0            0 1.0000E 00

     4 * 2    DELE
5.0000E-01             0
1.0000E 00             0
           0 5.0000E-01
           0 1.0000E 00

                CPTCON2 COMPLETE
```

SAMPLE NUMBER 10

```
     2 * 4    FP
-3.5581E-02                0                0                0
           0                0-3.5581E-02                0

     4 * 4    PHIPI
 1.0000E 00 1.0000E 01                0                0
           0 1.0000E 00                0                0
           0                0 1.0000E 00 1.0000E 01
           0                0                0 1.0000E 00

     2 * 4    FP*PHIPI
-3.5581E-02-3.5581E-01                0                0
           0                0-3.5581E-02-3.5581E-01

     2 * 4    FE
-7.1161E-03                0                0                0
           0                0-7.1161E-03                0

     4 * 4    PHIEI
 1.0000E 00 1.0000E 01                0                0
           0 1.0000E 00                0                0
           0                0 1.0000E 00 1.0000E 01
           0                0                0 1.0000E 00

     2 * 4    FE*PHIEI
-7.1161E-03-7.1161E-02                0                0
           0                0-7.1161E-03-7.1161E-02
```

SAMPLE NUMBER 9

2 * 4 FP

-4.3634E-02 0 0 0
0 0 -4.3634E-02 0

4 * 4 PHIPI

1.0000E 00 9.0000E 00 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 9.0000E 00
0 0 0 1.0000E 00

2 * 4 FP*PHIPI

-4.3634E-02 -3.9271E-01 0 0
0 0 -4.3634E-02 -3.9271E-01

2 * 4 FE

-8.7269E-03 0 0 0
0 0 -8.7269E-03 0

4 * 4 PHIEI

1.0000E 00 9.0000E 00 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 9.0000E 00
0 0 0 1.0000E 00

2 * 4 FE*PHIEI

-8.7269E-03 -7.8542E-02 0 0
0 0 -8.7269E-03 -7.8542E-02

SAMPLE NUMBER 8

2 * 4 FP

-5.4745E-02 0 0 0
0 0 -5.4745E-02 0

4 * 4 PHIPI

1.0000E 00 8.0000E 00 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 8.0000E 00
0 0 0 1.0000E 00

2 * 4 FP*PHIPI

-5.4745E-02 -4.3796E-01 0 0
0 0 -5.4745E-02 -4.3796E-01

2 * 4 FE

-1.0949E-02 0 0 0
0 0 -1.0949E-02 0

4 * 4 PHIEI

1.0000E 00 8.0000E 00 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 8.0000E 00
0 0 0 1.0000E 00

2 * 4 FE*PHIEI

-1.0949E-02 -8.7591E-02 0 0
0 0 -1.0949E-02 -8.7591E-02

SAMPLE NUMBER 7

2 * 4 FF

-7.0652E-02 0 0 0
0 0 -7.0652E-02 0

4 * 4 PHIPI

1.0000E 00 7.0000E 00 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 7.0000E 00
0 0 0 1.0000E 00

2 * 4 FF*PHIPI

-7.0652E-02 -4.9457E-01 0 0
0 0 -7.0652E-02 -4.9457E-01

2 * 4 FE

-1.4130E-02 0 0 0
0 0 -1.4130E-02 0

4 * 4 PHIEI

1.0000E 00 7.0000E 00 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 7.0000E 00
0 0 0 1.0000E 00

2 * 4 FE*PHIEI

-1.4130E-02 -9.8913E-02 0 0
0 0 -1.4130E-02 -9.8913E-02

SAMPLE NUMBER 6

2 * 4 FF

-9.4502E-02 0 0 0
0 0-9.4502E-02 0

4 * 4 PHIPI

1.0000E 00 6.0000E 00 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 6.0000E 00
0 0 0 1.0000E 00

2 * 4 FF*PHIPI

-9.4502E-02-5.6701E-01 0 0
0 0-9.4502E-02-5.6701E-01

2 * 4 FE

-1.8900E-02 0 0 0
0 0-1.8900E-02 0

4 * 4 PHIEI

1.0000E 00 6.0000E 00 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 6.0000E 00
0 0 0 1.0000E 00

2 * 4 FE*PHIEI

-1.8900E-02-1.1340E-01 0 0
0 0-1.8900E-02-1.1340E-01

SAMPLE NUMBER 5

2 * 4 FP

-1.3235E-01	0	0	0
0	0	-1.3235E-01	0

4 * 4 FPIPI

1.0000E 00	5.0000E 00	0	0
0	1.0000E 00	0	0
0	0	1.0000E 00	5.0000E 00
0	0	0	1.0000E 00

2 * 4 FP*PIPI

-1.3235E-01	-6.6176E-01	0	0
0	0	-1.3235E-01	-6.6176E-01

2 * 4 FE

-2.6471E-02	0	0	0
0	0	-2.6471E-02	0

4 * 4 FPIEI

1.0000E 00	5.0000E 00	0	0
0	1.0000E 00	0	0
0	0	1.0000E 00	5.0000E 00
0	0	0	1.0000E 00

2 * 4 FE*PIEI

-2.6471E-02	-1.3235E-01	0	0
0	0	-2.6471E-02	-1.3235E-01

SAMPLE NUMBER 4

2 * 4 FP

-1.9663E-01	0	0	0
0	0	0-1.9663E-01	0

4 * 4 PHIPI

1.0000E 00	4.0000E 00	0	0
0	1.0000E 00	0	0
0	0	0 1.0000E 00	4.0000E 00
0	0	0	1.0000E 00

2 * 4 FP*PHIPI

-1.9663E-01	-7.8652E-01	0	0
0	0	0-1.9663E-01	-7.8652E-01

2 * 4 FE

-3.9326E-02	0	0	0
0	0	0-3.9326E-02	0

4 * 4 PHIEI

1.0000E 00	4.0000E 00	0	0
0	1.0000E 00	0	0
0	0	0 1.0000E 00	4.0000E 00
0	0	0	1.0000E 00

2 * 4 FE*PHIEI

-3.9326E-02	-1.5730E-01	0	0
0	0	0-3.9326E-02	-1.5730E-01

SAMPLE NUMBER 3

2 * 4 FP

-3.125CE-01		0	0
0		0-3.125CE-01	0

4 * 4 PHIPI

1.0000E 00	3.0000E 00	0	0
0	1.0000E 00	0	0
0	0	1.0000E 00	3.0000E 00
0	0	0	1.0000E 00

2 * 4 FP*PHIPI

-3.125CE-01	-9.375CE-01	0	0
0	0	-3.125CE-01	-9.375CE-01

2 * 4 FE

-6.250E-02		0	0
0		0-6.250CE-02	0

4 * 4 PHIEI

1.0000E 00	3.0000E 00	0	0
0	1.0000E 00	0	0
0	0	1.0000E 00	3.0000E 00
0	0	0	1.0000E 00

2 * 4 FE*PHIEI

-6.250CE-02	-1.875CE-01	0	0
0	0	-6.250CE-02	-1.875CE-01

SAMPLE NUMBER 2

2 * 4 FP

-5.0000E-01 0 0 0
0 0-5.0000E-01 0

4 * 4 FPIPI

1.0000E 00 2.0000E 00 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 2.0000E 00
0 0 0 1.0000E 00

2 * 4 FP*PFIPI

-5.0000E-01-1.0000E 00 0 0
0 0-5.0000E-01-1.0000E 00

2 * 4 FE

-1.0000E-01 0 0 0
0 0-1.0000E-01 0

4 * 4 FFIEI

1.0000E 00 2.0000E 00 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 2.0000E 00
0 0 0 1.0000E 00

2 * 4 FE*PFIEI

-1.0000E-01-2.0000E-01 0 0
0 0-1.0000E-01-2.0000E-01

SAMPLE NUMBER 1

2 * 4 FP

-4.1667E-01	0	0	0
0	0	-4.1667E-01	0

4 * 4 PHIPI

1.00000E 00	1.00000E 00	0	0
0	1.00000E 00	0	0
0	0	1.00000E 00	1.00000E 00
0	0	0	1.00000E 00

2 * 4 FP*PHIPI

-4.1667E-01	-4.1667E-01	0	0
0	0	-4.1667E-01	-4.1667E-01

2 * 4 FE

-8.3333E-02	0	0	0
0	0	-8.3333E-02	0

4 * 4 PHIEI

1.00000E 00	1.00000E 00	0	0
0	1.00000E 00	0	0
0	0	1.00000E 00	1.00000E 00
0	0	0	1.00000E 00

2 * 4 FE*PHIEI

-8.3333E-02	-8.3333E-02	0	0
0	0	-8.3333E-02	-8.3333E-02

```
GRAPH TITLED
     FP(1,1) AND FE(1,1) VS TIME
     HUTCHISON,PERMENTER,628,OPTML.2 RUN 1     2 1C 67
HAS BEEN PLOTTED.



          TERMINAL PERFORMANCE DATA

          4 * 1    XF-XE
      -9.3765E-C2
       3.CCC2E C1
      -4.6882E-C2
       2.5C12E CC

                    PERFORMANCE INDEX DUE TC STATES            1.C95CE-C2

                    PERFORMANCE INDEX DUE TC CONTROL EFFORT    2.3430E C1

                    TOTAL PERFORMANCE INDEX                    2.3441E C1

GRAPH TITLED
     NORTH VS EAST,PURSUIT-EVASION TRAJECTORIES
     HUTCHISON,PERMENTER,628,OPTML.2 RUN 1     2 1C 67
HAS BEEN PLOTTED.



     4 MINUTES, 37 SECONDS.
END JOB 1CS.
```

```
1234567890123456789012345678901234567890123456789012345678901234567890
                    INPUT DATA FOR RUN 21 FOLLOWS
RUN 21
 2 10 67
.1          T
          50       ITAPE
          50       IFPFE
          50       IPURSUE
 404        AP
.0          1.0          .0           .0
.0          .0           1.0          .0
.0          .0           .0           1.0
-.15585     -.69454      -1.5633      -1.7593
 401        BP
.0
.0
.0
1.0
 404        AE
.0          .0           .0           .0
.0          .0           .0           .0
.0          .0           .0           .0
.0          .0           .0           .0
 400        BE
BLANK CARD HERE
BLANK CARD HERE
BLANK CARD HERE
BLANK CARD HERE
 404        Q
1000.       .0           .0           .0
.0          1000.        .0           .0
.0          .0           1000.        .0
.0          .0           .0           1000.
```

```
                                                                    1.
                                                                    1.
                                                                    1.
```

```
                INPUT DATA

                GRAPH IDENTIFIER  RUN 21
                GRAPH IDENTIFIER   2 10 67
                1.0000E-01  IS THE SAMPLED-DATA PERIOD
                  50  SAMPLES WILL BE GENERATED ON TAPE
                  50  SAMPLES WILL BE USED FOR FP AND FE PLOT
                  50  SAMPLES WILL BE USED FOR TRAJECTORIES PLOT

     4 * 4   AP
          0 1.0000E 00           0           0
          0           0 1.0000E 00           0
          0           0           0 1.0000E 00
-1.5585E-01-6.9454E-01-1.5633E 00-1.7593E 00

     4 * 1   PP
          0
          0
          0
 1.0000E 00

     4 * 4   AE
          0           0           0           0
          0           0           0           0
          0           0           0           0
          0           0           0           0

     4 * 0   BE



     4 * 4   C
 1.0000E 03           0           0           0
          0 1.0000E 03           0           0
          0           0 1.0000E 03           0
          0           0           0 1.0000E 03
```

```
     1 *  1    RP
1.0CCCE C0

     0 *  C    RE

     4 *  1    XP(ZERC)
1.0CCCE 02
1.0CCOE 01
          0
          C

     4 *  1    XE(ZERO)
4.0CCCE 01
2.CCCOE 01
1.0CCOE 01
          0
```

```
                    OUTPUT INFORMATION

     SUBROUTINE PHIDEL  NUM =   12     SIZE = 1.0000E-21     T = 1.0000E-01

       4 * 4    PHIP
 1.0000E 00 9.9997E-02 4.9937E-03 1.5946E-04
-2.4852E-05 9.9989E-01 9.9748E-02 4.7131E-03
-7.3454E-04-3.2983E-03 9.9252E-01 9.1456E-02
-1.4253E-02-6.4254E-02-1.4627E-01 8.3162E-01

       4 * 1    DELP
 4.0222E-06
 1.5946E-04
 4.7131E-03
 9.1456E-02

     SUBROUTINE PHIDEL  NUM =    1     SIZE = 1.0000E-21     T = 1.0000E-01

       4 * 4    PHIE
 1.0000E 00           0           0           0
           0 1.0000E 00           0           0
           0           0 1.0000E 00           0
           0           0           0 1.0000E 00

       4 * 0    DELE


                    OPTCON2 COMPLETE
```

SAMPLE NUMBER 5)

1 * 4 FP
-1.1468E CC 2.3437E 0C-1.8C8CE 00 5.413CE-01

4 * 4 PHIPI
4.3905E-C1 1.6644E 00 2.2489E 00 1.3382E CC
-2.0855E-C1-4.9C36E-01-4.2758E-01-1.C528E-01
1.64C8E-02-1.3543E-C1-3.2578E-C1-2.4237E-01
3.7773E-02 1.8474E-01 2.4346E-01 1.0062E-01

1 * 4 FP*PHIPI
-1.0C15E 00-2.7131E 0C-2.8605E 0C-1.2887E 0C

0 * 4 FE

4 * 4 PHIEI
1.CCC0E 00 C C C
C 1.CCCCE 00 0 0
0 C 1.0000E 00 0
0 C C 1.CCC0E 0C

C * 4 FE*PHIEI

SAMPLE NUMBER 49

1 * 4 FP
-1.2277E 00 2.4518E 0C-1.8468E 0C 5.3680E-01

4 * 4 PHIPI
4.5998E-C1 1.7127E 0C 2.29CCE 00 1.3475E CC
-2.1CC0E-01-4.7588E-01-3.9379E-C1-8.0549E-02
1.2554E-02-1.54C6E-01-3.4996E-01-2.52C8E-C1
3.9287E-C2 1.8764E-01 2.4CC3E-01 9.3530E-02

1 * 4 FP*PHIPI
-1.0817E 00-2.8842E 00-3.0018E C0-1.3360E CC

C * 4 FE

4 * 4 PHIEI
1.CCC0E C0 C C C
0 1.CCCCE CC C C
0 C 1.CCC0E 00 C
C C C 1.CCC0E 0C

C * 4 FE*PHIEI

SAMPLE NUMBER 46

1 * 4 FP
-1.3149E 00 2.5642E 0C-1.8841E 00 5.3075E-C1

4 * 4 PHIPI
4.81C4E-01 1.7595E 0C 2.3276E 00 1.3542E 0C
-2.11C6E-01-4.5954E-01-3.5761E-01-5.4886E-C2
8.5540E-03-1.7294E-01-3.7373E-01-2.61C4E-01
4.C684E-02 1.8986E-01 2.3515E-01 8.5523E-02

1 * 4 FP*PHIPI
-1.1662E 00-3.0653E 0C-3.1487E 00-1.3842E 0C

C * 4 FE

4 * 4 PHIEI
1.0CC0E 00 0 0 0
0 1.0C0CE 00 0 0
0 0 1.CCCCE 00 0
0 0 0 1.CC00E 00

C * 4 FE*PHIEI

SAMPLE NUMBER 47

1 * 4 FP
-1.4C89E 00 2.6EC7E 0C-1.9192E 00 5.2295E-01

4 * 4 PHIPI
5.0218E-01 1.8C45E 0C 2.3615E 00 1.3584E 00
-2.1171E-01-4.4129E-01-3.19C7E-01-2.8368E-02
4.4212E-03-1.92C1E-01-3.9694E-01-2.6916E-01
4.1948E-02 1.9136E-01 2.2877E-01 7.6588E-02

1 * 4 FP*PHIPI
-1.2616E 00-3.2569E 00-3.301CE C0-1.4333E 0C

C * 4 FE

4 * 4 PHIEI
1.0CC0E CC 0 0 0
0 1.0C0CE CC 0 0
0 0 1.0CCCF CC 0
0 0 0 1.0CC0E CC

C * 4 FE*PHIEI

SAMPLE NUMBER 44

1 * 4 FP

-1.5101E 00 2.8611E 00-1.9519E 00 5.1211E-01

4 * 4 PHIPI

5.2337E-01 1.8477E 00 2.3914E 00 1.3559E 00
-2.1194E-01-4.2113E-01-2.7824E-01-1.0857E-02
1.6921E-04-2.1118E-01-4.1943E-01-2.7633E-01
4.3066E-02 1.9209E-01 2.2080E-01 6.6719E-02

1 * 4 FP*PHIPI

-1.3622E 00-3.4592E 00-3.4588E 00-1.4832E 00

C * 4 FE

4 * 4 PHIEI

1.0000E 00 0 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 0
0 0 0 1.0000E 00

C * 4 FE*PHIEI

SAMPLE NUMBER 45

1 * 4 FP

-1.6190E 00 2.9247E 00-1.9803E 00 5.0115E-01

4 * 4 PHIPI

5.4455E-01 1.8887E 00 2.4171E 00 1.3586E 00
-2.1174E-01-3.9905E-01-2.3521E-01 2.6863E-02
-4.1866E-03-2.3040E-01-4.4105E-01-2.8247E-01
4.4023E-02 1.9200E-01 2.1119E-01 5.5901E-02

1 * 4 FP*PHIPI

-1.4706E 00-3.6725E 00-3.6220E 00-1.5337E 00

C * 4 FE

4 * 4 PHIEI

1.0000E 00 0 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 0
0 0 0 1.0000E 00

C * 4 FE*PHIEI

(Printout for sample periods 7 through 44 omitted.)

SAMPLE NUMBER 6

1 * 4 FP

-1.0729E 00-4.4468E CC-8.7346E 00 1.29C8E 00

4 * 4 PHIPI

9.9932E-01 5.9689E-01 1.7281E-01 2.7450E-02
-4.2781E-03 9.8026E-01 5.5398E-01 1.2451E-01
-1.94C5E-02-9.0757E-02 7.8561E-01 3.3492E-01
-5.2198E-02-2.52C2E-01-6.1434E-01 1.9638E-01

1 * 4 FP*PHIPI

-9.51C5E-01-4.532CE 0C-1.0304E 01-3.2551E 00

C * 4 FE

4 * 4 PHIEI

1.0C00E 00 0 0 0
0 1.0000E 0C 0 0
0 0 1.0000E 00 0
0 0 0 1.0000E 0C

C * 4 FE*PHIEI

SAMPLE NUMBER 5

1 * 4 FP

-6.8121E-01-3.6626E CC-1.0213E 01 7.6661E-01

4 * 4 PHIPI

9.9966E-01 4.9845E-01 1.2144E-01 1.6637E-02
-2.5928E-03 9.8811E-01 4.7244E-01 9.2167E-02
-1.4364E-02-6.6607E-02 8.44C2E-01 3.1C29E-01
-4.8359E-02-2.2988E-01-5.5169E-01 2.9812E-01

1 * 4 FP*PHIPI

-5.6185E-01-3.4545E 00-1.0856E 01-3.2895E CC

C * 4 FE

4 * 4 PHIEI

1.0CCCE 00 0 0 0
0 1.0CCCE 0C 0 0
0 0 1.00CCE 00 0
0 0 0 1.0CCCE 00

C * 4 FE*PHIEI

SAMPLE NUMBER 4

1 * 4 FP
-3.7197E-01-2.6909E 00-1.0679E 01-2.5390E-01

4 * 4 PHIPI
9.9986E-01 3.9935E-01 7.8500E-02 8.9171E-03
-1.3697E-03 9.9366E-01 3.8541E-01 6.2813E-02
-9.7893E-03-4.5016E-02 8.9547E-01 2.7490E-01
-4.2843E-02-2.0072E-01-4.7477E-01 4.1184E-01

1 * 4 FP*PHIPI
-2.5276E-01-2.2908E 00-1.0508E 01-3.2124E 00

0 * 4 FE

4 * 4 PHIEI
1.0000E 00 0 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 0
0 0 0 1.0000E 00

0 * 4 FE*PHIEI

SAMPLE NUMBER 3

1 * 4 FP
-1.5132E-01-1.5698E 00-9.2626E 00-1.9244E 00

4 * 4 PHIPI
9.9995E-01 2.9979E-01 4.4513E-02 3.9366E-03
-6.1352E-04 9.9722E-01 2.9363E-01 3.7587E-02
-5.8579E-03-2.6719E-02 9.3846E-01 2.2751E-01
-3.5457E-02-1.6387E-01-3.8238E-01 5.3821E-01

1 * 4 FP*PHIPI
-2.7859E-02-1.0479E 00-8.4244E 00-3.2026E 00

0 * 4 FE

4 * 4 PHIEI
1.0000E 00 0 0 0
0 1.0000E 00 0 0
0 0 1.0000E 00 0
0 0 0 1.0000E 00

0 * 4 FE*PHIEI

```
                SAMPLE NUMBER     2

     1 * 4    FP
-3.2440E-02-5.5481E-01-5.6219E 00-4.4776E 00

     4 * 4    PHIPI
 9.9999E-01 1.9996E-01 1.9901E-02 1.2201E-03
-1.9015E-04 9.9914E-01 1.9805E-01 1.7755E-02
-2.7671E-03-1.2521E-02 9.7139E-01 1.6681E-01
-2.5998E-02-1.1863E-01-2.7330E-01 6.7791E-01

     1 * 4    FP*PHIPI
 9.9630E-02 4.0730E-02-4.3478E 00-3.9831E 00

     C * 4    FE

     4 * 4    PHIEI
 1.0000F 00           0           0           0
           0 1.0000E 00           0           0
           0           0 1.0000E 00           0
           0           0           0 1.0000E 00

     C * 4    FE*PHIEI

                SAMPLE NUMBER     1


     1 * 4    FP
-4.2851E-04-1.6989E-02-5.0212E-01-9.7434E 00

     4 * 4    PHIPI
 1.0000E 00 9.9997E-02 4.9937E-03 1.5946E-04
-2.4852E-05 9.9989E-01 9.9748E-02 4.7131E-03
-7.3454E-04-3.2983E-03 9.9252E-01 9.1456E-02
-1.4253E-02-6.4254E-02-1.4627E-01 8.3162E-01

     1 * 4    FP*PHIPI
 1.3882E-01 6.1068E-01 9.2513E-01-8.1488E 00

     C * 4    FE

     4 * 4    PHIEI
 1.0000E 00           0           0           0
           0 1.0000E 00           0           0
           0           0 1.0000E 00           0
           0           0           0 1.0000E 00

     C * 4    FE*PHIEI
```

```
GRAPH TITLED
     FP(1,1) AND FE(1,1) VS TIME
     HUTCHISON,PERMENTER,628,OPTML.2 RUN 21    2 10 67
HAS BEEN PLOTTED.


          TERMINAL PERFORMANCE DATA

          4 * 1    XP-XE
       1.1185E 00
      -3.4193E 00
       2.3391E 00
      -1.4372E 00

                    PERFORMANCE INDEX DUE TO STATES            2.048CE 04

                    PERFORMANCE INDEX DUE TO CONTROL EFFORT    1.2816E 05

                    TOTAL PERFORMANCE INDEX                    1.4864E 05

GRAPH TITLED
     VELOCITY VS POSITION,PHASE PLANE,ONE-SIDED CASE
     HUTCHISON,PERMENTER,628,OPTML.2 RUN 21    2 10 67
HAS BEEN PLOTTED.

GRAPH TITLED
     XP(1,1),XP(2,1),XP(3,1) VS TIME,ONE-SIDED CASE
     HUTCHISON,PERMENTER,628,OPTML.2 RUN 21    2 10 67
HAS BEEN PLOTTED.

ALL POINTS HAVE THE SAME COORDINATES.
NO FURTHER GRAPH OUTPUT UNTIL MODCURV NEXT IS ZERO OR ONE.


     7 MINUTES, 59 SECONDS.
END JOB 138.
```

APPENDIX C

SAMPLE VARIATIONS
OF
BASIC COMPUTER PROGRAM

```
-COOP,628,PERMENTER,I/22/3/22/49/3/15/2S/E/45=54,20,20000,0,OPTIMAL2.
-FTN,L,E.
      PROGRAM OPTIMAL2
      DIMENSION TIME(100),POSITN(100),VELOCTY(100),ACCELER(100)
C         THIS IS A GENERAL PROGRAM USED TO CONDUCT COMPUTER
C         SIMULATION TESTS OF THE STRATEGY DERIVED IN THE
C         THESIS-
C                   AN OPTIMAL PURSUIT-EVASION STRATEGY
C                   FOR LINEAR SAMPLED-DATA SYSTEMS
C                   BY C.H. HUTCHISON AND L.F. PERMENTER
C         NOTATION -
C         T         SAMPLED-DATA PERIOD
C         ITAPE     NO. OF SAMPLE PERIODS FOR WHICH STRATEGY
C                   IS TO BE RECORDED ON MAGNETIC TAPE
C         IFPFE     NO. OF SAMPLE PERIODS FOR WHICH ELEMENTS
C                   OF FP AND FE MATRICES ARE TO BE PLOTTED
C         IPURSUE   NO. OF SAMPLE PERIODS FOR WHICH SIMULATED
C                   TRAJECTORIES ARE TO BE RUN
C         NP        ORDER OF THE PURSUIT PLANT
C         MP        DIMENSION OF PURSUIT PLANT CONTROL VECTOR
C         XP        PURSUIT PLANT STATE VECTOR
C         UP        PURSUIT PLANT CONTROL VECTOR
C         AP AND BP (NP*NP) AND (NP*MP) MATRICES DEFINED BY
C                   THE CONSTANT COEFFICIENT MATRIX
C                   DIFFERENTIAL EQUATION-
C                             XPDOT = AP*XP + BP*UP
C         PHIP      (NP*NP) CONSTANT COEFFICIENT STATE
C                   TRANSITION MATRIX FOR PURSUIT PLANT
C         DELP      (NP*MP) CONSTANT COEFFICIENT DISTRIBUTION
C                   MATRIX FOR PURSUIT PLANT
C         PHIPI     PHIP**I
C         FP        (MP*NP) PURSUIT PLANT VARIABLE COEFFICIENT
C                   FEEDBACK GAIN MATRIX
C         RP      (MP*MP) PURSUIT CONTROL WEIGHTING MATRIX
C         SIMILAR NOTATION IS USED FOR THE EVASION PLANT
```

```
C          Z          FINAL STATEMENT DIFFERENCE VECTOR IF NO
C                     CONTROLS APPLIED
C                     (NXN) FINAL RATE WEIGHTING MATRIX
      DIMENSION AP(10,10),AA(10,10),BB(10,10),CC(10,10),
     1PHIP(10,10),PHIE(10,10),DELP(10,10),DELE(10,10),
     2PHIPI(10,10),PHIEI(10,10),XP(10,10),XE(10,10),U(10,10),
     3RF(10,10),RE(10,10),A1(10,10),A2(10,10),FP(10,10),
     4FE(10,10),Z(10,10),UP(10,10),UE(10,10),IT(12),
     5XPNORTH(900),XENORTH(900),XPEAST(900),XEEAST(900),
     6FPD(900),FED(900),TAU(900)
      REWIND 22
      PRINT 560
  560 FORMAT(1H1,20X,10HINPUT DATA,/)
      READ 561,IT(11)
      READ 561,IT(12)
  561 FORMAT(A8)
      PRINT 562,IT(11),IT(12)
  562 FORMAT(/,20X,18HGRAPH IDENTIFIER  ,A8)
      READ 515,T
  515 FORMAT(E10.4)
      PRINT 516,T
  516 FORMAT(/,20X,E10.4,2X,26HIS THE SAMPLED-DATA PERIOD)
      READ 510,ITAPE
      READ 510,IFPFE
      READ 510,IPURSUE
  510 FORMAT(I10)
      IF(ITAPE-IPURSUE)518,517,517
  517 IF(ITAPE-IFPFE)518,519,519
  518 PRINT 514
  514 FORMAT(/,20X,47HITAPE MUST BE ATLEAST EQUAL TO IFPFE OR IPURSUE)
      GO TO 999
  519 PRINT 511,ITAPE
  511 FORMAT(/,20X,I4,2X,33HSAMPLES WILL BE GENERATED ON TAPE)
      PRINT 512,IFPFE
  512 FORMAT(/,20X,I4,2X,39HSAMPLES WILL BE USED FOR FP AND FE PLOT)
```

```
      PRINT 513,IPUNCH
  513 FORMAT(/,20X,I5,2X,42HSAMPLES WILL BE USED FOR TRAJECTORIES PLOT)
      CALL MAURICE(AP,NP,NP)
      NA=8HAP
      CALL JOHANN(AP,NP,NP,NA)
      CALL MAURICE(BP,NP,MP)
      NA=8HBP
      CALL JOHANN(BP,NP,MP,NA)
      CALL MAURICE(AE,NE,NE)
      NA=8HAE
      CALL JOHANN(AE,NE,NE,NA)
      CALL MAURICE(BE,NE,ME)
      NA=8HBE
      CALL JOHANN(BE,NE,ME,NA)
      CALL MAURICE(Q,NP,NP)
      NA = 8HQ
      CALL JOHANN(Q,NP,NP,NA)
      CALL MAURICE(RP,MP,MP)
      NA = 8HRP
      CALL JOHANN(RP,MP,MP,NA)
      CALL MAURICE(RE,ME,ME)
      NA = 8HRE
      CALL JOHANN(RE,ME,ME,NA)
      CALL MAURICE(XP,NP,1)
      NA=8HXP(ZERO)
      CALL JOHANN(XP,NP,1,NA)
      CALL MAURICE(XE,NE,1)
      NA = 8HXE(ZERO)
      CALL JOHANN(XE,NE,1,NA)
C          INPUT AND RECORDING OF DATA CARDS AND PLANT PARAMETERS
C          IS NOW COMPLETE
      PRINT 501
  501 FORMAT(1H1,20X,18HOUTPUT INFORMATION,/)
      CALL PHIDEL(AP,BP,NP,MP,T,PHIP,DELP)
      NA=8HPHIP
```

```
      CALL JOHANN(PHIP,NP,MP,NA)
      NA=8HDELP
      CALL JOHANN(DELP,MP,MP,NA)
      CALL PHIDEL(AL,DE,NE,IE,I,P  ,  ELE)
      NA=8HPHIE
      CALL JOHANN(PHIE,NE,NE,NA)
      NA=8HDELE
      CALL JOHANN(DELE,NE,ME,NA)
      N = NE
  520 CALL OPTCON2(PHIP,PHIE,DELP,DELE,J,RP,RE,ITAPE,N,MP,ME)
  521 J=IFPFE-1
      DO 522 I=1,J
      CALL LOCATE2(FP,MP,N,K)
      CALL LOCATE2(FE,ME,N,K)
      CALL LOCATE2(PHIPI,N,N,K)
  522 CALL LOCATE2(PHIEI,N,N,K)
      DO 100 I=1,IFPFE
      CALL LOCATE2(FP,MP,N,K)
      CALL LOCATE2(FE,ME,N,K)
      CALL LOCATE2(PHIPI,N,N,K)
      CALL LOCATE2(PHIEI,N,N,K)
      IBACK=IFPFE-I+1
      PRINT 571,IBACK
  571 FORMAT(/,20X,15HSAMPLE NUMBER  ,I3,/)
      NA = 8HFP
      CALL JOHANN(FP,MP,N,NA)
      NA = 8HPHIPI
      CALL JOHANN(PHIPI,N,N,NA)
      CALL PROD(FP,PHIPI,A1,MP,N,N)
      NA=8HFP*PHIPI
      CALL JOHANN(A1,MP,N,NA)
      NA = 8HFE
      CALL JOHANN(FE,ME,N,NA)
      NA = 8HPHIEI
      CALL JOHANN(PHIEI,N,N,NA)
```

```
      CALL PROJ(FE,PHIEI,A1,ME,N,N)
      HA=8HFE*PHIEI
      CALL JOHAN2(A1,ME,A,A)
      IF(K-1) 506,506,504
  504 DO 505 J=1,8
  505 BACKSPACE 22
  506 T1=I
      TAU(I)=T*T1
      FED(I)=FE(1,1)
      FPD(I)=FP(1,1)
  100 CONTINUE
      LAST = 0
      IT(1)=8HFP(1,1)
      IT(2)=8HAND FE(1
      IT(3)=8H,1) VS T
      IT(4)=8HIME
      IT(5)=8H
      IT(6)=8H
      IT(7) =8HHUTCHISO
      IT(8) =8HN,PERMEN
      IT(9) =8HTER,628,
      IT(10)=8HOPTML.2
      LABEL=4HFP
      CALL DRAW(IFPFE,TAU,FPD,1,0,LABEL,IT,0,0,0,0,1,1,7,7,0,LAST)
      LABEL=4H  FE
      CALL DRAW(IFPFE,TAU,FED,3,0,LABEL,IT,0,0,0,0,1,1,7,7,0,LAST)
      REWIND 22
      COSTR = 0.0
  554 J=IPURSUE-1
      DO 601 I=1,J
      CALL LOCATE2(FP,MP,NP,K)
      CALL LOCATE2(FE,ME,NE,K)
      CALL LOCATE2(PHIPI,NP,NP,K)
      CALL LOCATE2(PHIEI,NE,NE,K)
  601 CONTINUE
```

```
C          *** VARIOUS CTRL MULT CHANGES MAY BE NEEDED
C          *** BELOW FOR DIFFERENT PLANTS
           XPNORM(1)=XP(1,1)
           XPLAST(1)=XP(3,1)
           XENORM(1)=XE(1,1)
           XELAST(1)=XE(3,1)
C              * * * * PROGRAM CHANGE 20N 21 * * * *
           TIME(1)=0.0
           POSITN(1)=XP(1,1)
           VELOCTY(1)=XP(2,1)
           ACCELER(1)=XP(3,1)
C              * * * * END CHANGE * * * *
           JPURSUE=IPURSUE+1
           DO 602 INDEX=2,JPURSUE
C              * * * * PROGRAM CHANGE 20N 21 * * * *
           T3=INDEX
           TIME(INDEX)=T*T3
           POSITN(INDEX)=XP(1,1)
           VELOCTY(INDEX)=XP(2,1)
           ACCELER(INDEX)=XP(3,1)
C              * * * * END CHANGE * * * *
           CALL LOCATE2(FP,MP,NP,K)
           CALL LOCATE2(FE,ME,NE,K)
           CALL LOCATE2(PHIPI,NP,NP,K)
           CALL LOCATE2(PHIEI,NE,NE,K)
           IF(K-1)603,603,604
       604 DO 605 I=1,8
       605 BACKSPACE 22
       603 CALL PROD(PHIPI,XP,Z,NP,NP,1)
           CALL PROD(PHIEI,XE,A1,NE,NE,1)
           CALL SUBTRAC(Z,A1,Z,NE,1)
           CALL PROD(FP,Z,UP,MP,NP,1)
           CALL PROD(FE,Z,UE,ME,NE,1)
C              *** INSERT STATEMENTS HERE TO PROVIDE DIFFERENT
C              *** CONTROL STRATEGIES FOR PURSUIT OR EVADER
```

```
      CALL PLANT(PHIP,DELP,XP,UP,NP,MP)
      CALL PLANT(PHIE,DELE,XE,UE,NE,ME)
      XPNORTH(INDEX)=XP(1,1)
      XPEAST(INDEX)=XP(3,1)
      XENORTH(INDEX)=XE(1,1)
      XEEAST(INDEX)=XE(3,1)
      CALL TRANS(UP,A1,MP,1)
      CALL PROD(A1,RP,A2,1,MP,MP)
      CALL PROD(A2,UP,A1,1,MP,1)
      COSTR=COSTR+A1(1,1)
      CALL TRANS(UE,A1,ME,1)
      CALL PROD(A1,RE,A2,1,ME,ME)
      CALL PROD(A2,UE,A1,1,ME,1)
      COSTR=COSTR-A1(1,1)
  602 CONTINUE
      PRINT 200
  200 FORMAT(1H1,10X,25HTERMINAL PERFORMANCE DATA,/)
      CALL SUBTRAC(XP,XE,Z,NP,1)
      NA = 8HXP-XE
      CALL JOHANN(Z,NP,1,NA)
      CALL TRANS(Z,A1,NP,1)
      CALL PROD(A1,Q,A2,1,NP,NP)
      CALL PROD(A2,Z,A1,1,NP,1)
      PRINT 201, A1(1,1)
  201 FORMAT(/,20X,41HPERFORMANCE INDEX DUE TO STATES          ,E11.4,/)
      PRINT 202,COSTR
  202 FORMAT(/,20X,41HPERFORMANCE INDEX DUE TO CONTROL EFFORT  ,E11.4,/)
      TOTAL=A1(1,1)+COSTR
      PRINT 203,TOTAL
  203 FORMAT(/,20X,41HTOTAL PERFORMANCE INDEX                  ,E11.4,/)
C         * * * * PROGRAM CHANGE RUN 21 * * * *
      LABEL=4H
      IT(1)=8HVELOCITY
      IT(2)=8H VS POSI
      IT(3)=8HTION,PHA
```

```
      IT(4)=8HCASE PLOT
      IT(5)=8H,ONE-SID
      IT(6)=8HED CASE
      SCALE=20.
      CALL DRAW(JPURSUE,POSITN,VELOCTY,3,0,LABEL,IT,SCALE,SCALE,0,0,
     12,0,7,7,0,LAST)
      IT(1)=8HXP(1,1),
      IT(2)=8HXP(2,1),
      IT(3)=8HXP(3,1)
      IT(4)=8HVS TIME,
      IT(5)=8HONE-SIDE
      IT(6)=8HD CASE
      LAST=0
      SCALE=40.
      LABEL=4HP
      CALL DRAW(JPURSUE,TIME,POSITN,1,0,LABEL,IT,0,SCALE,3,0,
     12,0,7,7,0,LAST)
      LABEL=4HV
      CALL DRAW(JPURSUE,TIME,VELOCTY,2,0,LABEL,IT,0,SCALE,3,0,
     12,0,7,7,0,LAST)
      LABEL=4HA
      CALL DRAW(JPURSUE,TIME,ACCELER,3,0,LABEL,IT,0,SCALE,3,0,
     12,0,7,7,0,LAST)
      LAST=0
C          * * * * END CHANGE * * * *
      LAST = 0
      IT(1)=8HNORTH VS
      IT(2)=8H EAST,PU
      IT(3)=8HRSUIT-EV
      IT(4)=8HASION TR
      IT(5)=8HAJECTORI
      IT(6)=8HES
      SCALE = 100.0
      LABEL=4H  E
      CALL DRAW(JPURSUE,XEAST,XENORTH,1,0,LABEL,IT,SCALE,SCALE,0,0,
```

```
     1C,.,7,7,.,LAST)
      LABEL=4HP
      CALL DRAW(PURPOSE,XPEAT,YPNORTH,3.[ ,LABEL,IT,SCALE,SCALE,0,C,
     1,.[ ,7,7,.,LAST)
999  REWIND 22
      END
```

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	20
2.	Library Naval Postgraduate School Monterey, California 93940	2
3.	Commander, Naval Ordnance Systems Command (ORD-911) Department of the Navy Washington, D. C. 20360	1
4.	Commander, Naval Air Systems Command Department of the Navy Washington, D. C. 20360	1
5.	Professor Donald E. Kirk Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
6.	Professor Harold A. Titus Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
7.	LT Charles H. Hutchison, USN 592452/1100 1045 Vernon Ave. Winston-Salem, North Carolina 27106	1
8.	LT Lawrence F. Permenter, USN 630838/1310 100 Lakeview Drive Spartanburg, South Carolina 29302	1
9.	Mr. Scott R. Neal Naval Ordnance Test Station China Lake, California 93556	1

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)

Naval Postgraduate School
Monterey, California 93940

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP

3. REPORT TITLE

An Optimal Pursuit-Evasion Strategy for Linear Sampled-Data Systems

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Masters thesis - June 1967

5. AUTHOR(S) (Last name, first name, initial)

Hutchison, Charles H. LT, U. S. Navy 592452/1100
Permenter, Lawrence F. LT, U. S. Navy 630838/1310

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
June 1967	117	7

8a. CONTRACT OR GRANT NO.

b. PROJECT NO.

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S)

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10. AVAILABILITY/LIMITATION NOTICES This document is subject to special export controls; each transmittal to foreign government or foreign nationals may be made only with prior approval of the U. S. Naval Postgraduate School.

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY
1. Commander, NOSC (ORD-911)
2. Commander, NASC, Washington, D.C.

13. ABSTRACT

Differential game theory and dynamic programming are applied to derive the optimal strategy, or sequence of controls, for a class of linear, sampled-data, pursuit-evasion problems. The necessary and sufficient conditions for existence of the solution are derived. A digital computer program for simulating control generation and system trajectories is given. The results of simulation tests using this digital computer model are presented to demonstrate the salient characteristics of the strategy.

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Automatic gain control						
Control sequences						
Control simulators						
Control systems						
Dynamic programming						
Game theory						
Iterative methods						
Optimization						
Pursuit-evasion						